Chapter 7: The Primal-Dual Method

(cp. Williamson & Shmoys, Chapter 7)

Set Cover Problem

Given: A set of elements $E = \{e_1, \dots, e_n\}$, a family of subsets $\{S_1, \dots, S_m\} \subseteq 2^E$, and a weight $w_j \geq 0$ for each $j \in \{1, \dots, m\}$.

Task: Find $I \subseteq \{1, \dots, m\}$ minimizing $\sum_{j \in I} w_j$ such that $\bigcup_{j \in I} S_j = E$.

LP relaxation:

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} w_j \cdot x_j \\ \text{s.t.} \quad & \sum_{j : e_i \in S_j} x_j \geq 1 && \text{for all } i = 1, \dots, n \\ & x_j \geq 0 && \text{for all } j = 1, \dots, m \end{aligned}$$

Dual LP:

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} y_i \\ \text{s.t.} \quad & \sum_{e_i \in S_j} y_i \leq w_j && \text{for all } j = 1, \dots, m \\ & y_i \geq 0 && \text{for all } i = 1, \dots, n \end{aligned}$$

Primal-Dual Algorithm for Set Cover Problem (see Ch. 1)

1. set $y :\equiv 0$ and $I := \emptyset$;
2. while $\exists e_k \notin \bigcup_{j \in I} S_j$
3. increase y_k until $\exists j$ with $e_k \in S_j$ such that $\sum_{i : e_i \in S_j} y_i = w_j$;
4. set $I := I \cup \{j\}$;

Theorem ??

The primal-dual algorithm is an f-approximation algorithm for the Set Cover Problem where $f := \max_{i=1,\ldots,n} |\{j \mid e_i \in S_j\}|$.

Proof:

$$\sum_{j \in I} w_j = \sum_{j \in I} \sum_{i : e_i \in S_j} y_i = \sum_{i=1}^{n} y_i \cdot |\{j \in I \mid e_i \in S_j\}|$$
$$\leq f \cdot \sum_{i=1}^{n} y_i \leq f \cdot \mathrm{OPT}_{LP} \leq f \cdot \mathrm{OPT}$$

□

Approximate Complementary Slackness

Remark. The pair of feasible solutions (x, y) to the primal and dual LP found by the algorithm satisfies

$$x_j > 0 \implies \sum_{e_i \in S_j} y_i = w_j \qquad (compl.\ slackness)$$
$$y_i > 0 \implies \sum_{j : e_i \in S_j} x_j \leq f \qquad (approx.\ compl.\ slackness)$$

The analysis on the previous slide only relies on these two properties!

Feedback Vertex Set Problem

Given: Undirected graph $G = (V, E)$ with node weights $w_i \geq 0$, $i \in V$.

Task: Find $S \subseteq V$ minimizing $\sum_{i \in S} w_i$ such that $G[V \setminus S]$ is acyclic.

Integer programming formulation: Let $\mathcal{C}$ denote the set of all cycles in G.

$$\begin{aligned} \min \quad & \sum_{i \in V} w_i \cdot x_i & \\ \text{s.t.} \quad & \sum_{i \in C} x_i \geq 1 & \text{for all } C \in \mathcal{C}, \\ & x_i \in \{0, 1\} & \text{for all } i \in V. \end{aligned}$$

Dual of LP relaxation ($x \geq 0$):

$$\begin{aligned} \max \quad & \sum_{C \in \mathcal{C}} y_C & \\ \text{s.t.} \quad & \sum_{C \in \mathcal{C} :\, i \in C} y_C \leq w_i & \text{for all } i \in V, \\ & y_C \geq 0 & \text{for all } C \in \mathcal{C}. \end{aligned}$$

Primal-Dual Algorithm for Feedback Vertex Set Problem

1. set $y :\equiv 0$ and $S := \emptyset$;
2. while there is a cycle C in G
3. increase y_C until there is an $i \in C$ with $\sum_{C' : i \in C'} y_{C'} = w_i$;
4. set $S := S \cup \{i\}$ and delete i from G;
5. repeatedly remove nodes of degree one from G;

Analysis:

$$\sum_{i \in S} w_i = \sum_{i \in S} \sum_{C :\, i \in C} y_C = \sum_{C \in \mathcal{C}} |S \cap C| \cdot y_C$$

- Idea: If $|S \cap C| \leq \alpha$ whenever $y_C > 0$, we get performance ratio α.
- But: If we choose arbitrary C in each iteration, $|S \cap C|$ can be large.
- Idea: Always choose short cycle C with $|C| \leq \alpha$.
- But: This is not always possible (e. g., if graph is one large cycle).
- Idea: From path of nodes of degree two, algorithm chooses ≤ 1 node.

Refined Primal-Dual Algorithm for Feedback Vertex Set

Lemma 7.1.

In any graph G that has no nodes of degree one, there is a cycle with $\leq 2\lceil \log_2 n \rceil$ nodes of degree 3 or more, and it can be found in linear time.

Proof:... □

Theorem 7.2.

If the primal-dual algorithm chooses in each iteration a cycle with at most $\leq 2\lceil \log_2 n \rceil$ nodes of degree 3 or more, it has performance ratio $4\lceil \log_2 n \rceil$.

Proof:... □

Remarks.

- The LP relaxation has an integrality gap of $\Omega(\log n)$.
- There is a primal-dual 2-approximation algorithm based on a more sophisticated integer programming formulation

Shortest s-t-Path Problem

Given: Undir. graph $G = (V, E)$ with edge costs $c_e \geq 0$, $e \in E$; $s, t \in V$

Task: Find minimum-cost s-t-path.

IP formulation: (let $\mathcal{S} := \{S \subseteq V \mid s \in S,\ t \in V \setminus S\}$)

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e & \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1 & \text{for all } S \in \mathcal{S}, \\ & x_e \in \{0, 1\} & \text{for all } e \in E. \end{aligned}$$

Dual of LP relaxation ($x \geq 0$):

$$\begin{aligned} \max \quad & \sum_{S \in \mathcal{S}} y_S & \\ \text{s.t.} \quad & \sum_{S \in \mathcal{S}:\, e \in \delta(S)} y_S \leq c_e & \text{for all } e \in E, \\ & y_S \geq 0 & \text{for all } S \in \mathcal{S}. \end{aligned}$$

Primal-Dual Algorithm for Shortest s-t-Path Problem

1. set $y :\equiv 0$ and $F := \emptyset$;
2. while there is no s-t-path in F
3.   let C be the connected component of (V, F) containing s;
4.   increase y_C until there is an $e \in \delta(C)$ with $\sum_{S \in \mathcal{S}\colon e \in \delta(S)} y_S = c_e$;
5.   set $F := F \cup \{e\}$;
6. delete edges from F that do not lie on s-t-path in F;

Lemma 7.3.
Throughout the algorithm, the set of edges in F always forms a tree containing node s. □

Theorem 7.4.
The algorithm finds a shortest s-t-path.

Proof:. . . □

Steiner Forest Problem

Given: Graph $G = (V, E)$ with costs $c_e \geq 0$, $e \in E$; k pairs $s_i, t_i \in V$.

Task: Find $F \subseteq E$ minimizing $c(F)$ and connecting s_i and t_i, for all i.

IP formulation: (let $\mathcal{S}_i := \{S \subseteq V \mid |S \cap \{s_i, t_i\}| = 1\}$ and $\mathcal{S} := \bigcup_{i=1}^{k} \mathcal{S}_i$)

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e \cdot x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1 && \text{for all } S \in \mathcal{S}, \\ & x_e \in \{0, 1\} && \text{for all } e \in E. \end{aligned}$$

Dual of LP relaxation ($x \geq 0$):

$$\begin{aligned} \max \quad & \sum_{S \in \mathcal{S}} y_S \\ \text{s.t.} \quad & \sum_{S \in \mathcal{S}\colon e \in \delta(S)} y_S \leq c_e && \text{for all } e \in E, \\ & y_S \geq 0 && \text{for all } S \in \mathcal{S}. \end{aligned}$$

Primal-Dual Algorithm for Steiner Forest Problem

1. set $y :\equiv 0$ and $F := \emptyset$;
2. while not all s_i-t_i pairs are connected in (V, F)
3. let C connected comp. of (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i;
4. increase y_C until there is an $e \in \delta(C)$ with $\sum_{S \in \mathcal{S}:\, e \in \delta(S)} y_S = c_e$;
5. set $F := F \cup \{e\}$;

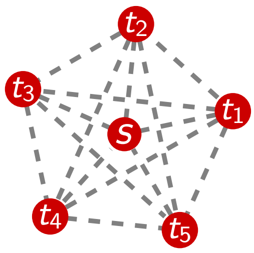
Analysis:

$$\sum_{e \in F} c_e = \sum_{e \in F} \sum_{S \in \mathcal{S}:\, e \in \delta(S)} y_S = \sum_{S \in \mathcal{S}} |\delta(S) \cap F| \cdot y_S$$

Problem:

It can happen that $|\delta(S) \cap F| = k$ for $y_S > 0$ and $\sum_{S \in \mathcal{S}} y_S \leq \frac{1}{k}\mathrm{OPT}$:

- $G = K_{k+1}$ (complete graph)
- $s_i := s$ for $i = 1, \ldots, k$
- $c_e := 1$ for all $e \in E$



Primal-Dual Algorithm for Steiner Forest Problem

1. set $y :\equiv 0$ and $F := \emptyset$;
2. while not all s_i-t_i pairs are connected in (V, F)
3. let C connected comp. of (V, F) with $|C \cap \{s_i, t_i\}| = 1$ for some i;
4. increase y_C until there is an $e \in \delta(C)$ with $\sum_{S \in \mathcal{S}:\, e \in \delta(S)} y_S = c_e$;
5. set $F := F \cup \{e\}$;

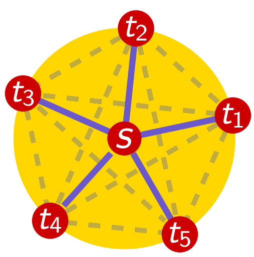
Analysis:

$$\sum_{e \in F} c_e = \sum_{e \in F} \sum_{S \in \mathcal{S}:\, e \in \delta(S)} y_S = \sum_{S \in \mathcal{S}} |\delta(S) \cap F| \cdot y_S$$

Problem:

It can happen that $|\delta(S) \cap F| = k$ for $y_S > 0$ and $\sum_{S \in \mathcal{S}} y_S \leq \frac{1}{k}\mathrm{OPT}$:

- $G = K_{k+1}$ (complete graph)
- $s_i := s$ for $i = 1, \ldots, k$
- $c_e := 1$ for all $e \in E$



- $y_{\{s\}} = 1$
- $|\delta(\{s\}) \cap F| = k$
- $\sum_S y_S = 1$, $\mathrm{OPT} = k$

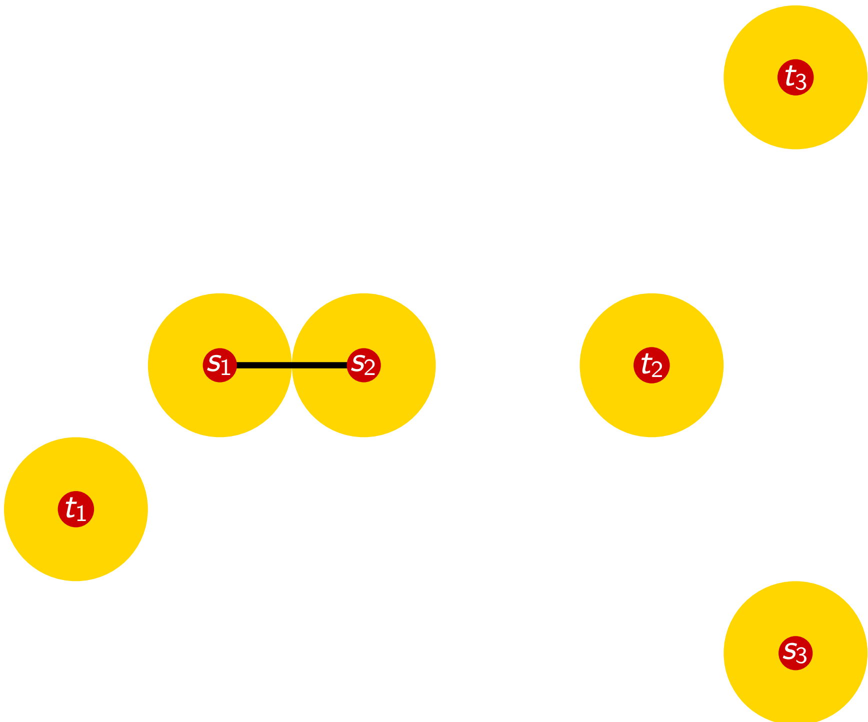
Refined Primal-Dual Algorithm for Steiner Forest Problem

set $y :\equiv 0$ and $F := \emptyset$;
while not all s_i-t_i pairs are connected in (V, F)
    let $\mathcal{C} := \{\text{conn. comp. } C \text{ of } (V, F)\colon |C \cap \{s_i, t_i\}| = 1 \text{ for some } i\}$;
    increase y_C for all $C \in \mathcal{C}$ uniformly until for some $e \in \delta(C)$, $C \in \mathcal{C}$

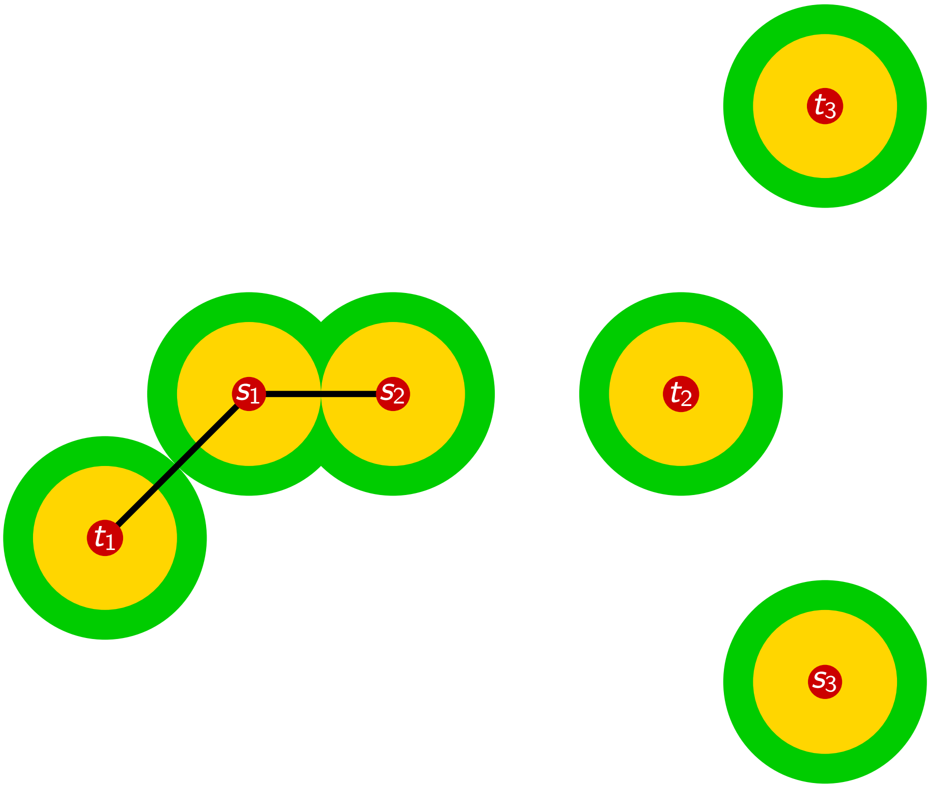
$$\sum_{S \in \mathcal{S}\colon e \in \delta(S)} y_S = c_e \ ;$$

    set $F := F \cup \{e\}$;
consider edges $e \in F$ in reverse of the order in which they were added
    if $F \setminus \{e\}$ is a feasible solution, then remove e from F;

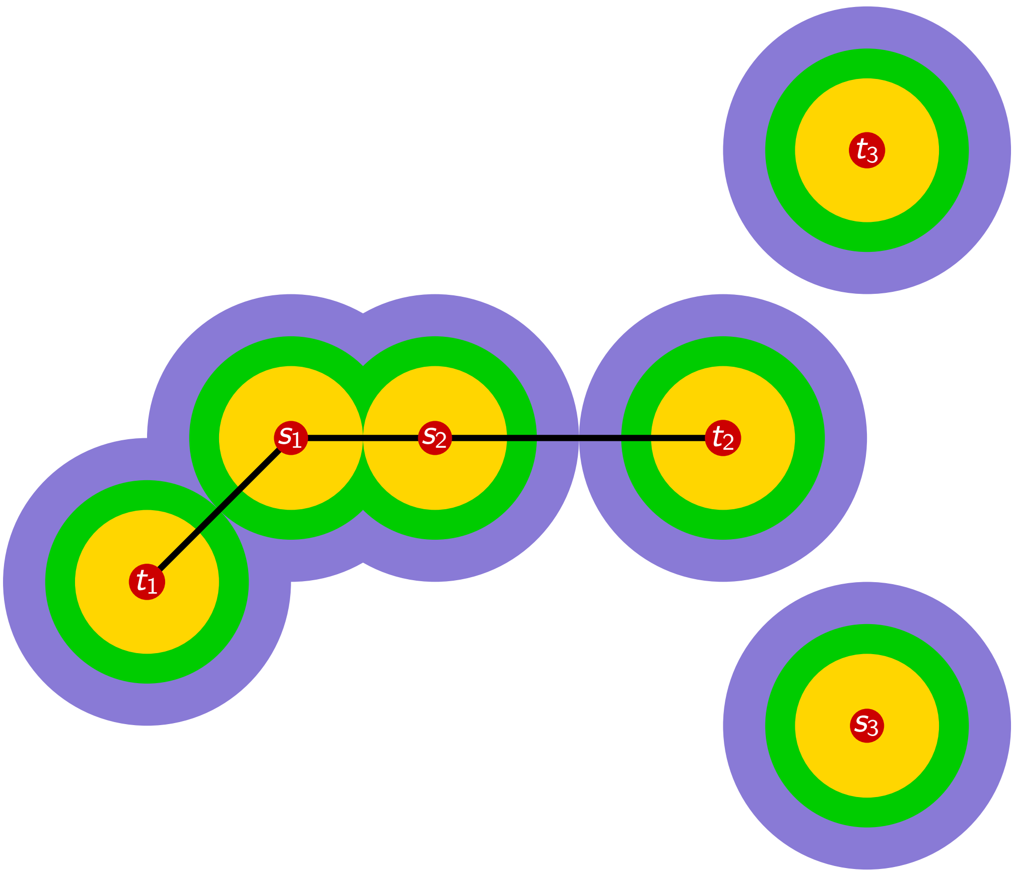
Example



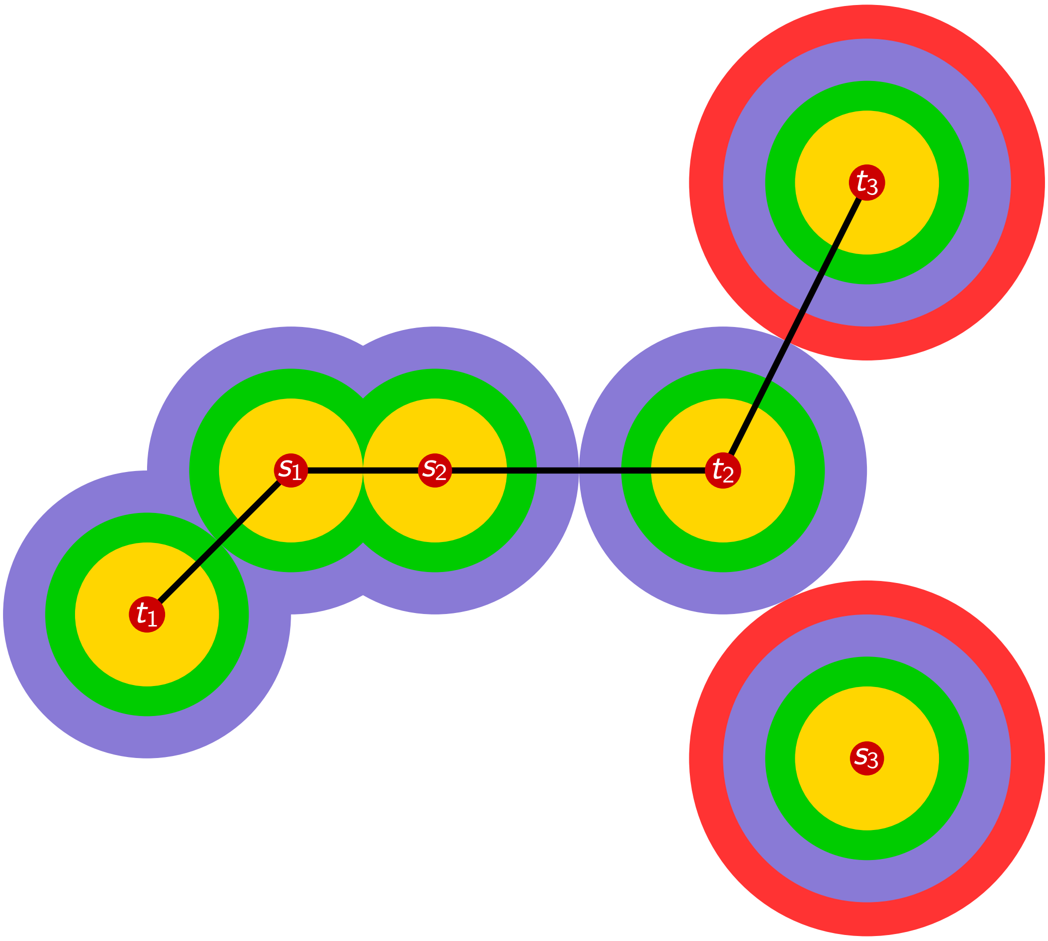
Example



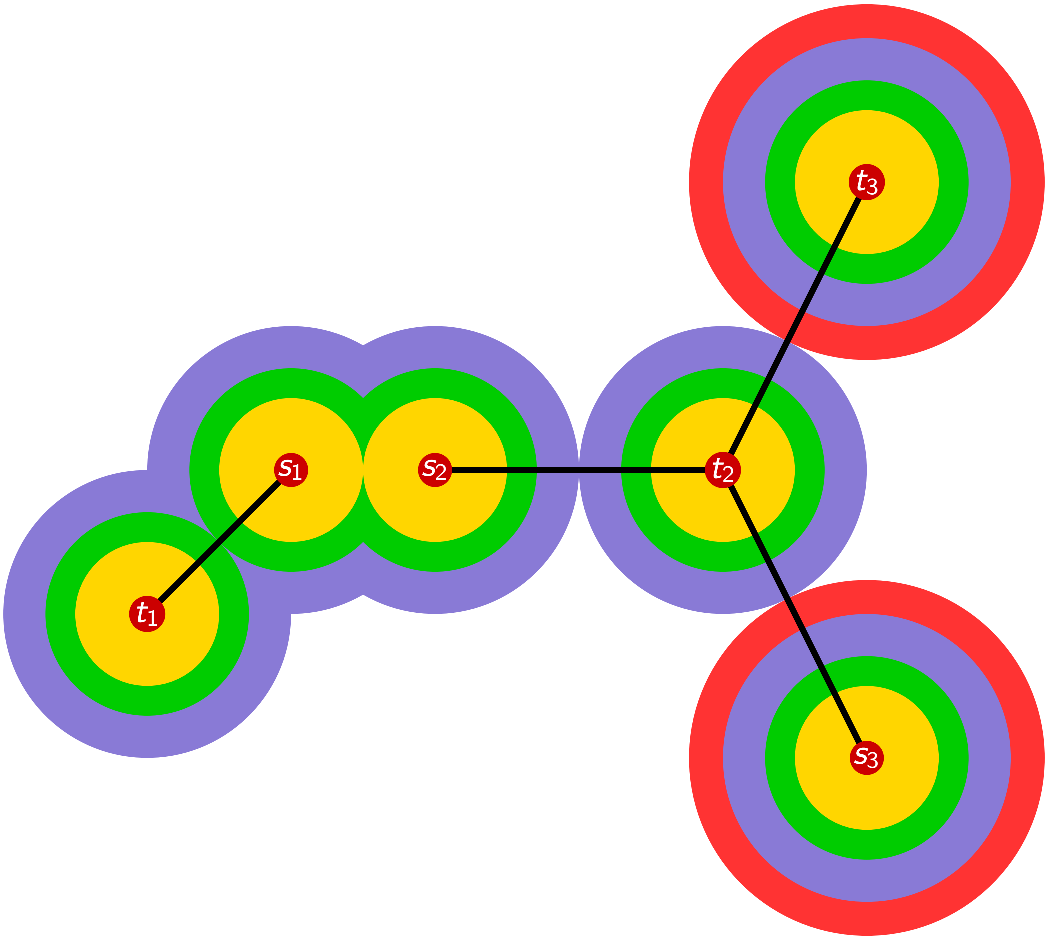
Example



Example



Example



113

Analysis

Observation. At any point in the algorithm, the set of edges F is a forest.

Lemma 7.5.

Let F' be the final set of edges returned by the algorithm. For any $\mathcal{C}$ in any iteration of the algorithm,

$$\sum_{C \in \mathcal{C}} |\delta(C) \cap F'| \leq 2|\mathcal{C}| \ .$$

Proof:. . . □

Theorem 7.6.

The refined primal-dual algorithm is a 2-approximation algorithm for the generalized Steiner Tree Problem.

Proof:. . . □

Corollary 7.7.

Integrality gap of the LP relaxation is at most 2. This bound is tight. □

Minimum Knapsack Problem

Given: n items with value $v_i \geq 0$ and size $s_i \geq 0$, $i = 1, \ldots, n$; demand D.

Task: Find $X \subseteq \{1, \ldots, n\}$ minimizing $\sum_{i \in X} s_i$ subject to $\sum_{i \in X} v_i \geq D$.

IP formulation:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} s_i \cdot x_i \\ \text{s.t.} \quad & \sum_{i=1}^{n} v_i \cdot x_i \geq D \\ & x_i \in \{0, 1\} \qquad \text{for all } i = 1, \ldots, n. \end{aligned}$$

Remarks.

- Replace $x_i \in \{0, 1\}$ with $0 \leq x_i \leq 1$ to obtain LP relaxation.
- The integrality gap of the LP relaxation is unbounded!
  Bad instance: Let $n = 2$, $v_1 = D - 1$, $v_2 = D$, $s_1 = 0$, $s_2 = 1$.

Stronger LP Relaxation

Notation: Let $[n] := \{1, \ldots, n\}$;
for $A \subseteq [n]$ and $i \in [n] \setminus A$ let $v_i^A := \min\{v_i, D - v(A)\}$.

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} s_i \cdot x_i \\
\text{s.t.} \quad & \sum_{i \in [n] \setminus A} v_i^A \cdot x_i \geq D - v(A) && \text{for all } A \subseteq [n], \\
& x_i \geq 0 && \text{for all } i \in [n].
\end{aligned}
$$

Dual LP:

$$
\begin{aligned}
\max \quad & \sum_{A \subseteq [n]} (D - v(A)) \cdot y_A \\
\text{s.t.} \quad & \sum_{A \subseteq [n] \colon i \notin A} v_i^A \cdot y_A \leq s_i && \text{for all } i \in [n], \\
& y_A \geq 0 && \text{for all } A \subseteq [n].
\end{aligned}
$$

Primal-Dual Algorithm for Minimum Knapsack Problem

1. set $y :\equiv 0$ and $X := \emptyset$;
2. while $v(X) < D$
3. increase y_X until for some $i \notin X$, $\sum_{A \subseteq [n] \colon i \notin A} v_i^A \cdot y_A = s_i$;
4. set $X := X \cup \{i\}$;

Theorem 7.8.
This is a 2-approximation algorithm for the Minimum Knapsack Problem.

Proof:... □

Corollary 7.9.
The integrality gap of the strengthened LP relaxation is at most 2. □

Uncapacitated Facility Location Problem

Given: Set of facilities F with opening costs $f_i \geq 0$, $i \in F$;
set of clients D with metric connection costs $c_{ij} \geq 0$, $i \in F$, $j \in D$.

Task: Choose $F' \subseteq F$ and assign each client to nearest facility in F'.

Objective: Minimize $\sum_{i \in F'} f_i + \sum_{j \in D} \min_{i \in F'} c_{ij}$.

IP formulation:

$$\begin{aligned} \min \quad & \sum_{i \in F} f_i \cdot y_i + \sum_{i \in F, j \in D} c_{ij} \cdot x_{ij} & \\ \text{s.t.} \quad & \sum_{i \in F} x_{ij} = 1 & \text{for all } j \in D, \\ & y_i - x_{ij} \geq 0 & \text{for all } i \in F, j \in D, \\ & x_{ij}, y_i \in \{0, 1\} & \text{for all } i \in F, j \in D. \end{aligned}$$

LP Relaxation and Dual LP

primal LP:

$$\begin{aligned} \min \quad & \sum_{i \in F} f_i \cdot y_i + \sum_{i \in F, j \in D} c_{ij} \cdot x_{ij} & \\ \text{s.t.} \quad & \sum_{i \in F} x_{ij} = 1 & \text{for all } j \in D, \\ & y_i - x_{ij} \geq 0 & \text{for all } i \in F, j \in D, \\ & x_{ij}, y_i \geq 0 & \text{for all } i \in F, j \in D. \end{aligned}$$

dual LP:

$$\begin{aligned} \max \quad & \sum_{j \in D} v_j & \\ \text{s.t.} \quad & \sum_{j \in D} w_{ij} \leq f_i & \text{for all } i \in F, \\ & v_j - w_{ij} \leq c_{ij} & \text{for all } i \in F, j \in D, \\ & w_{ij} \geq 0 & \text{for all } i \in F, j \in D. \end{aligned}$$

Interpretation of dual LP:

- v_j is total amount that client j wants to pay for being served.
- client j might contribute w_{ij} to facility i for being connected to i.

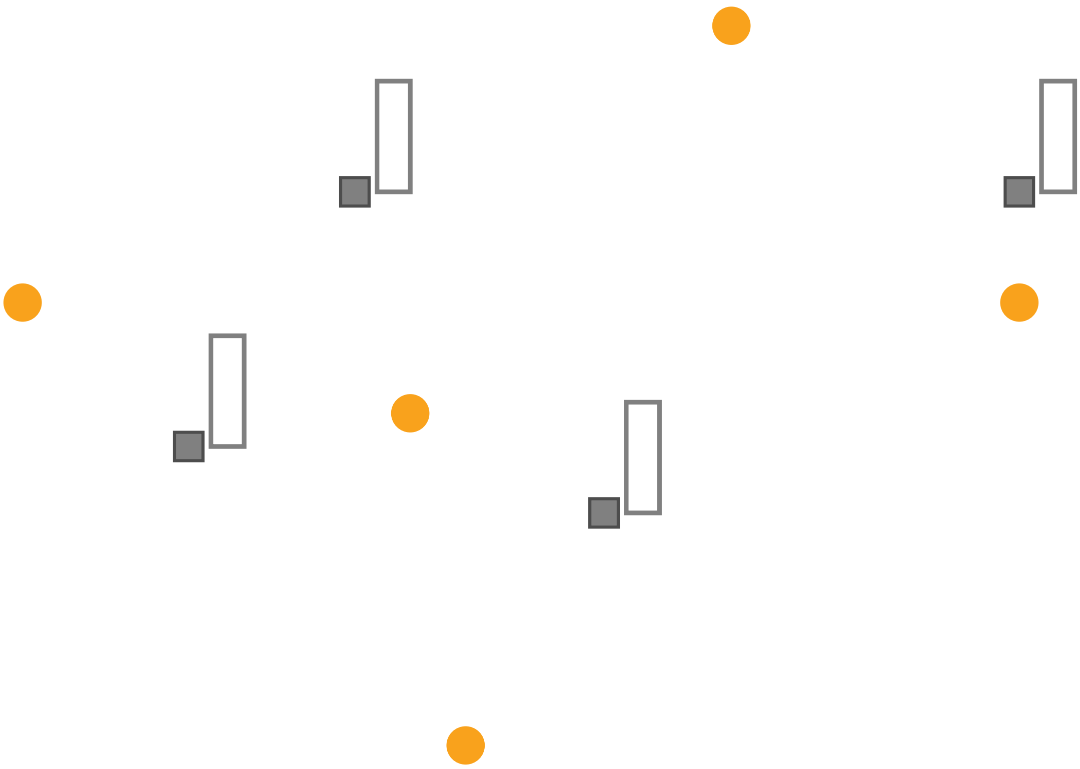
Primal-Dual Algorithm for Uncapacitated Facility Location

Notation: For the current feasible dual solution (v, w) and $j \in D$ let

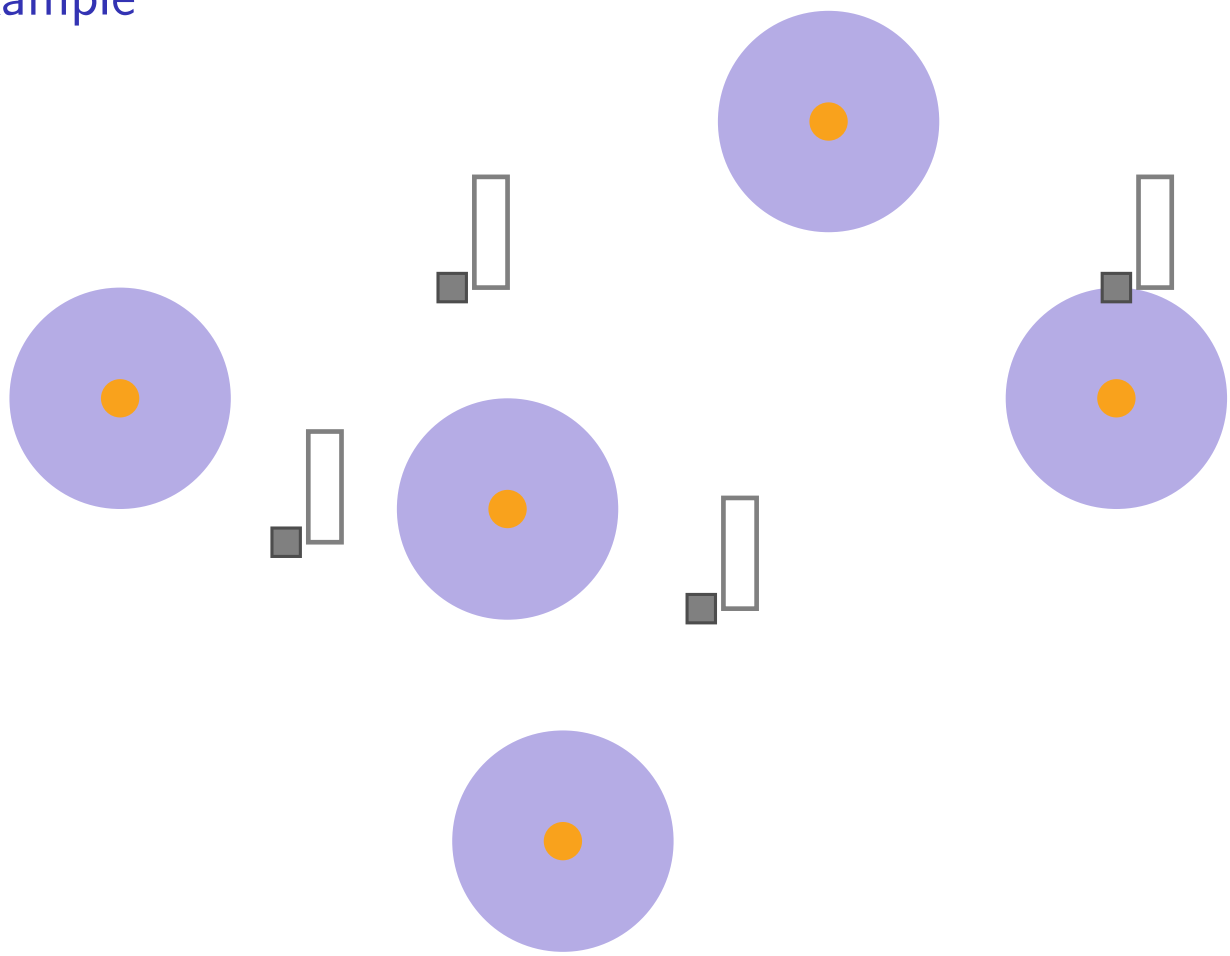
$$N(j) := \{i \in F \mid v_j \geq c_{ij}\} \ .$$

1. set $S := D$, $F'' := \emptyset$, $v_j := 0$, $w_{ij} := 0$ for all $i \in F$, $j \in D$;
2. while $S \neq \emptyset$
3. for all $j \in S$ increase v_j and w_{ij} for all $i \in N(j)$ uniformly until $N(j) \cap F'' \neq \emptyset$ for some $j \in S$ or $\sum_{j \in D} w_{ij} = f_i$ for some $i \notin F''$;
4. if $\sum_{j \in D} w_{ij} = f_i$ for some $i \notin F''$, then $F'' := F'' \cup \{i\}$;
5. if $N(j) \cap F'' \neq \emptyset$ for some $j \in S$, then $S := S \setminus \{j\}$;
6. set $F' := \emptyset$
7. while $F'' \neq \emptyset$ pick $i \in F''$;
8. $F' := F' \cup \{i\}$; $F'' := F'' \setminus \{h \mid \exists j \in D,\ w_{ij} > 0,\ w_{hj} > 0\}$;
9. open all facilities in F'; connect each $j \in D$ to nearest $i \in F'$;

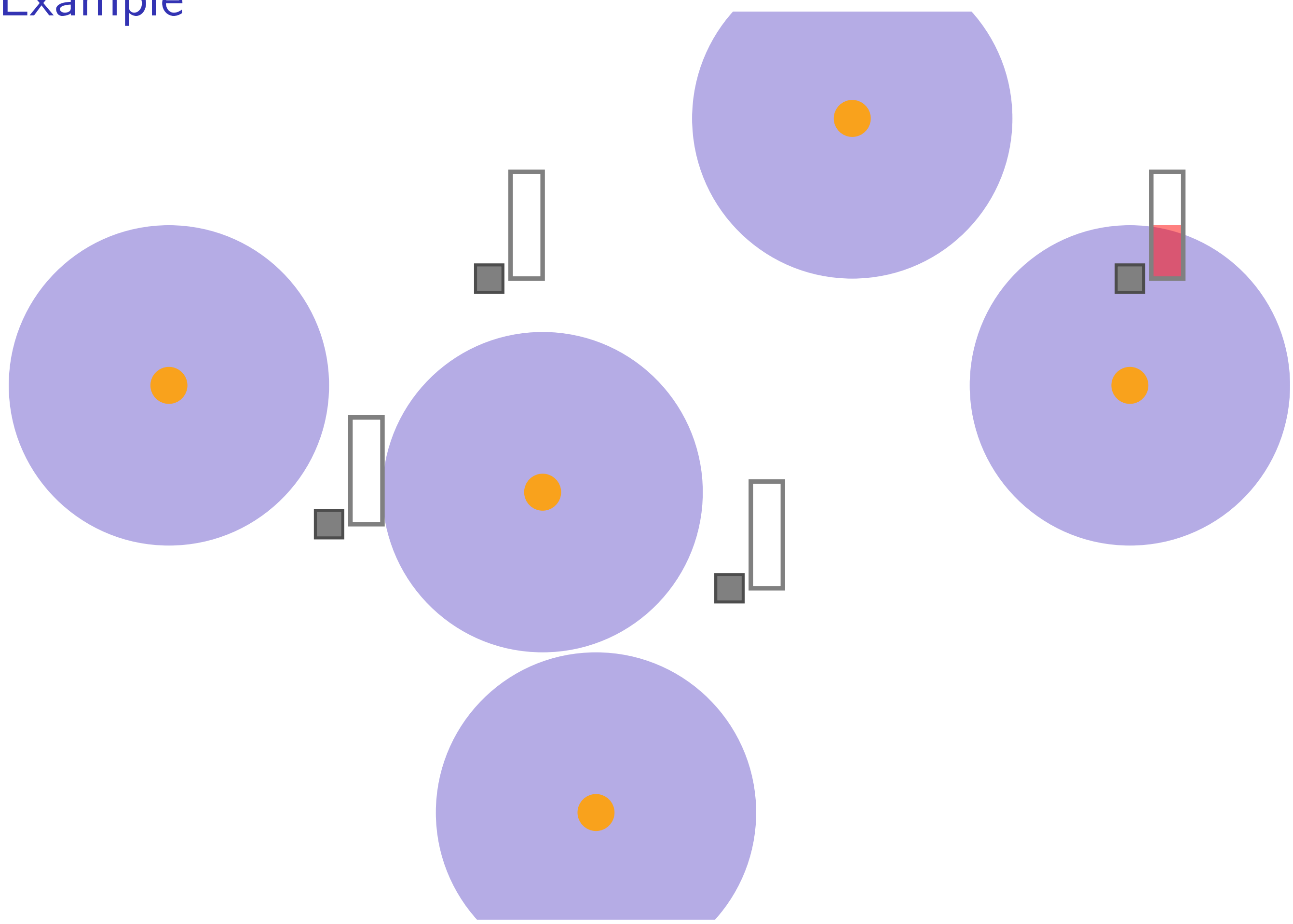
Example



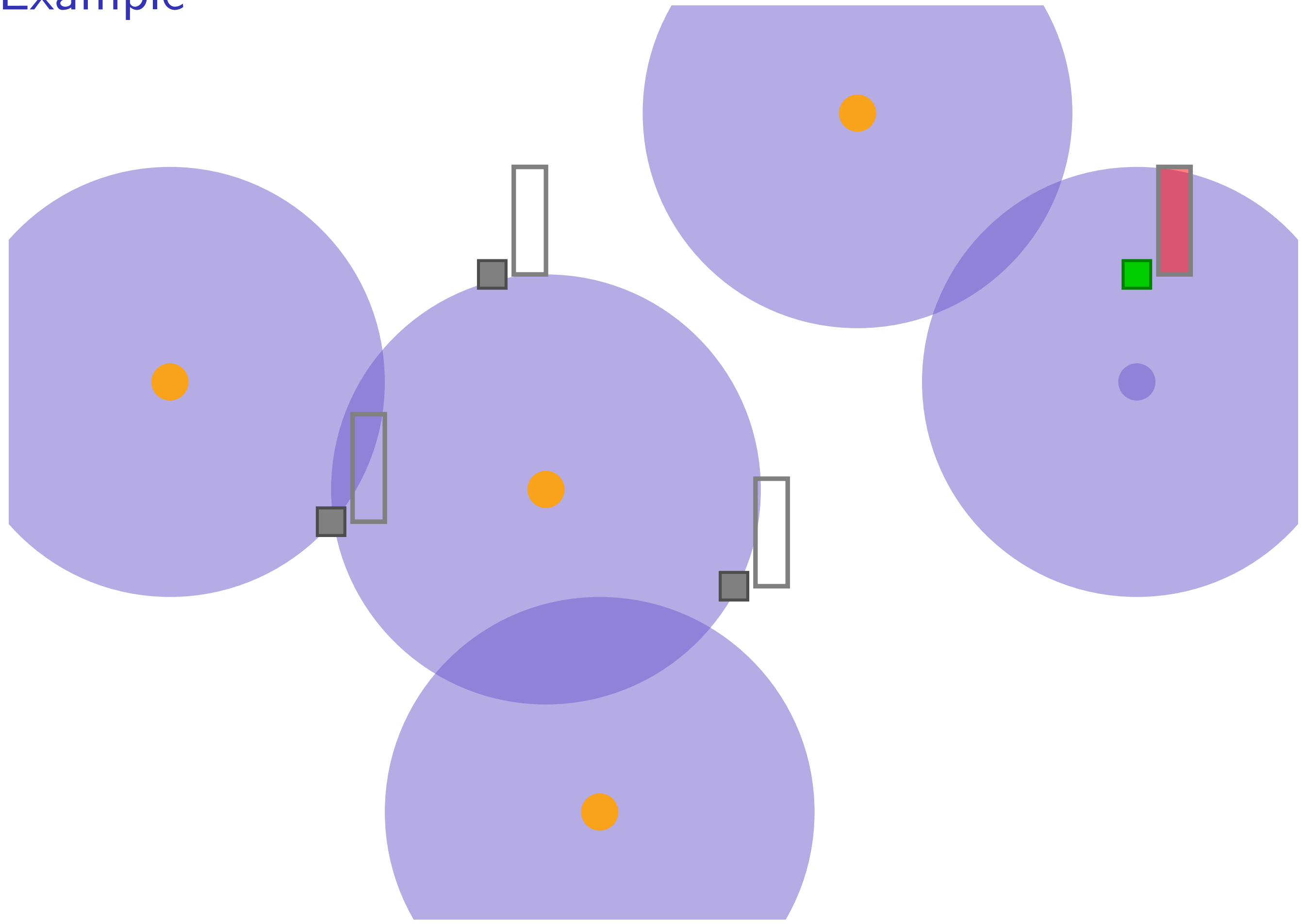
Example



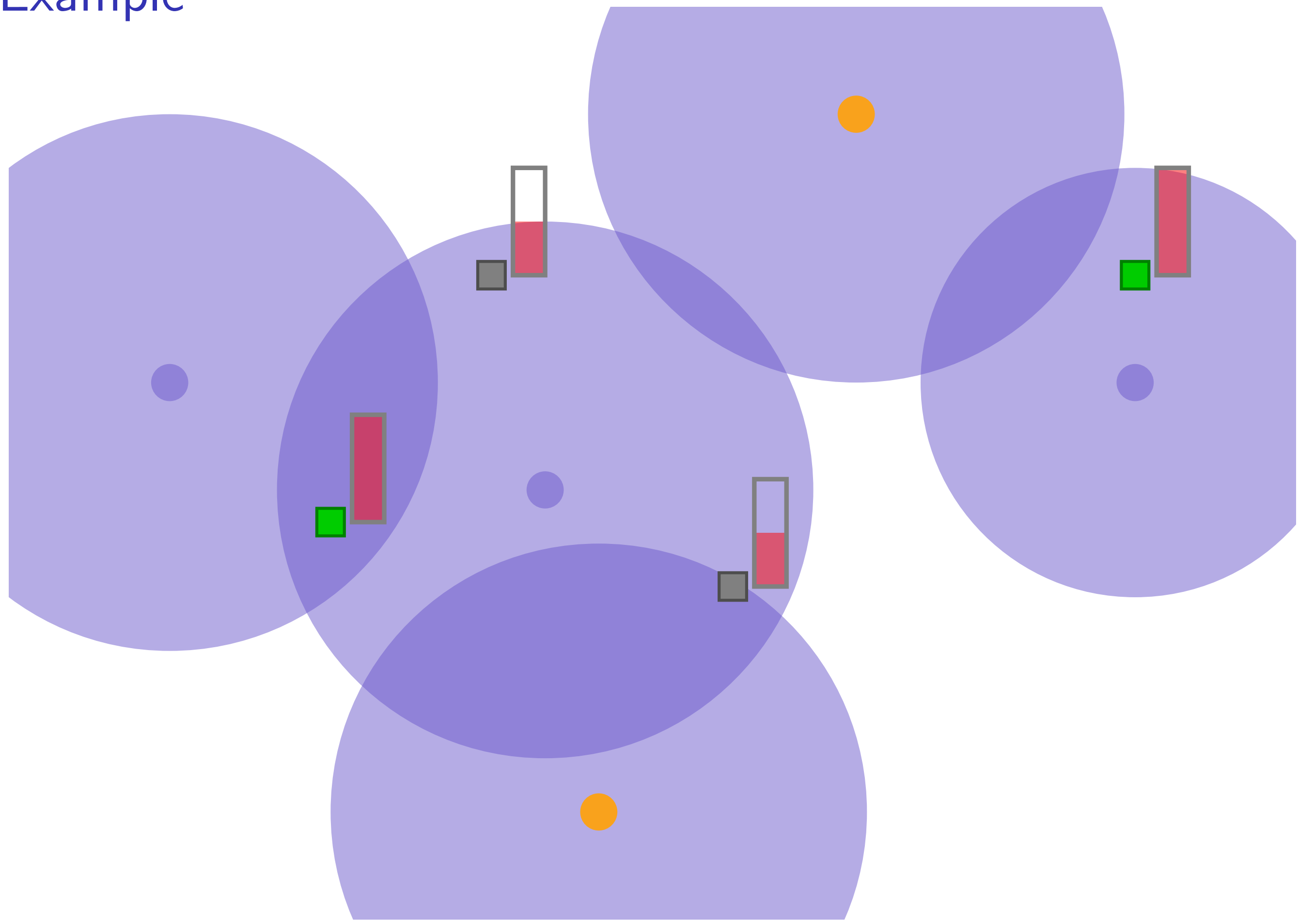
Example



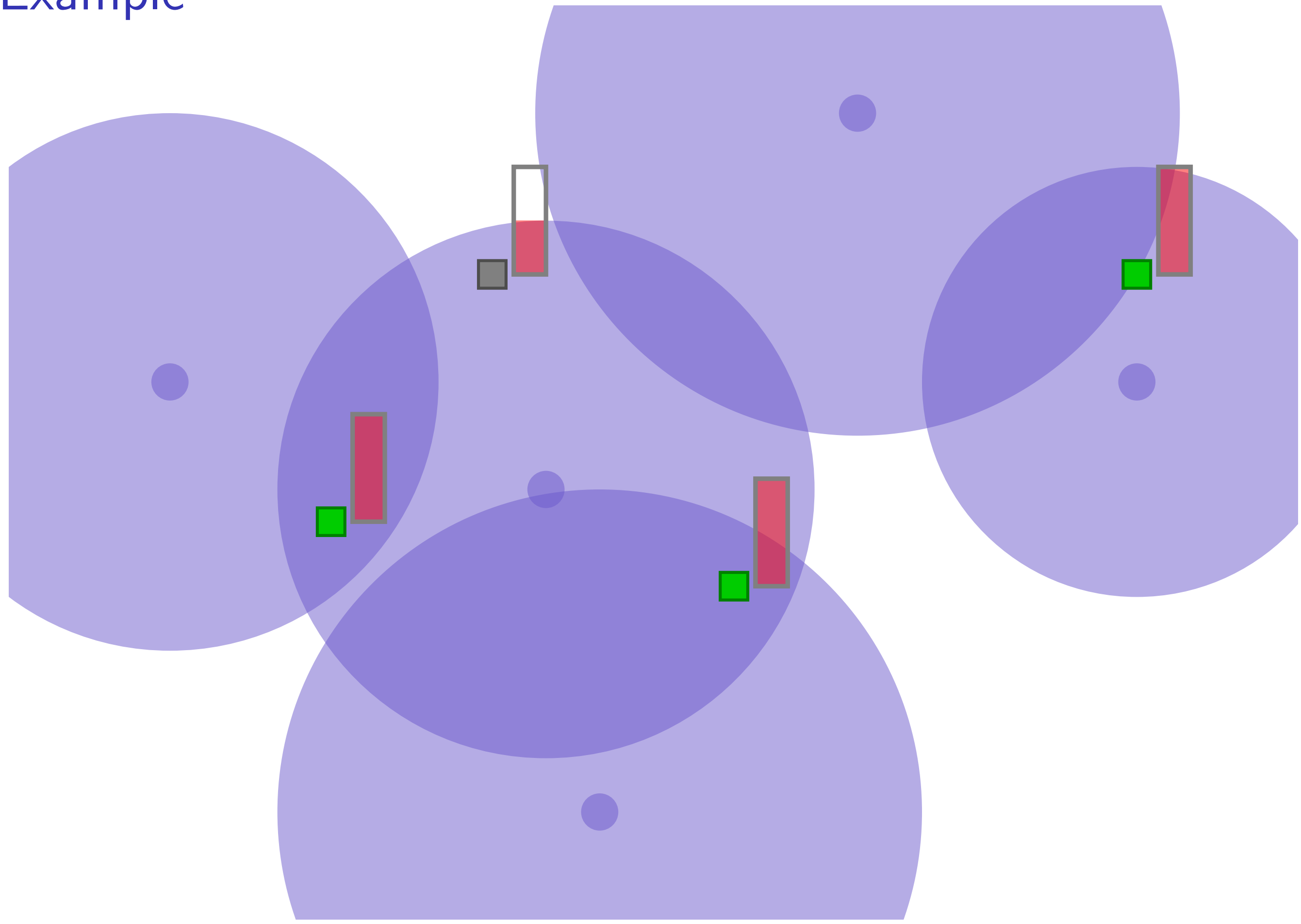
Example



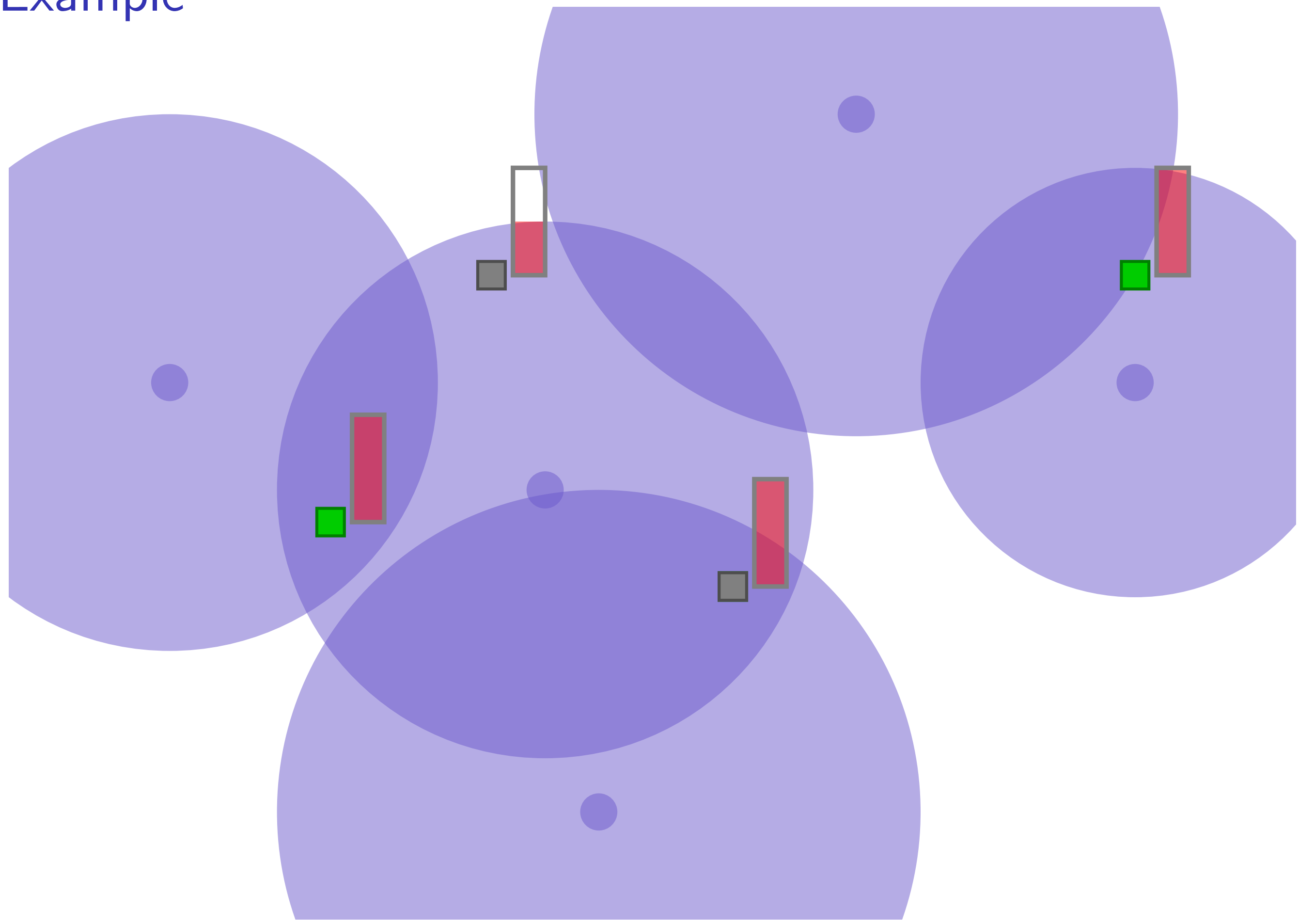
Example



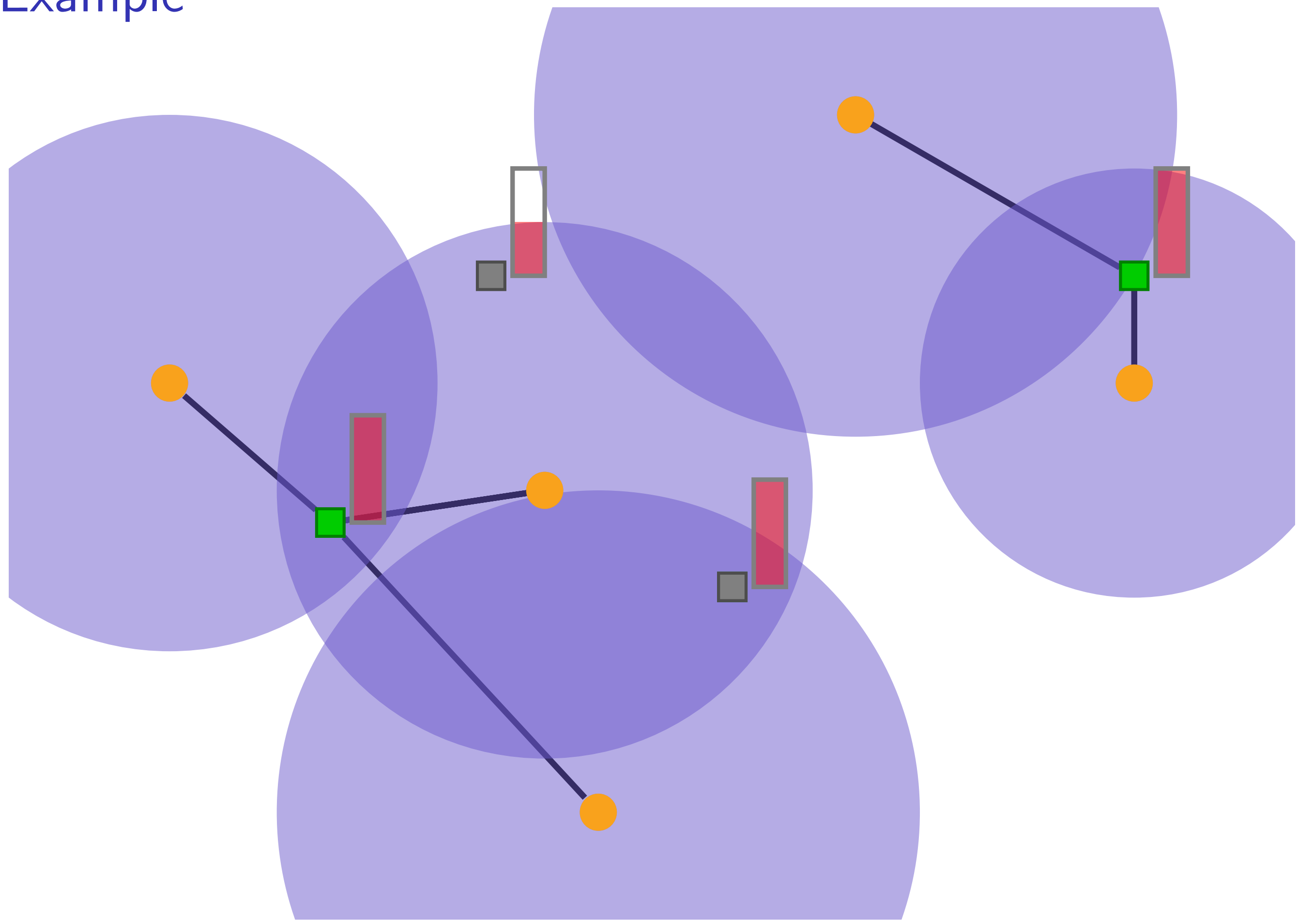
Example



Example



Example



Analysis

Lemma 7.10.

If a client j does not have a neighbor in F', then there is a facility $i \in F'$ such that $c_{ij} \leq 3v_j$.

Proof:. . . □

Theorem 7.11.

The algorithm is a 3-approximation algorithm for the uncapacitated facility location problem.

Proof:. . . □

k-Median Problem

Given: Positive integer k; set of facilities F; set of clients D; metric connection costs $c_{ij} \geq 0$, $i \in F$, $j \in D$.

Task: Find $S \subseteq F$ with $|S| \leq k$ minimizing $\sum_{j \in D} \min_{i \in S} c_{ij}$.

IP formulation:

$$
\begin{aligned}
\mathrm{OPT}_k \quad := \quad \min \quad & \sum_{i \in F, j \in D} c_{ij} \cdot x_{ij} & \\
\text{s.t.} \quad & \sum_{i \in F} x_{ij} = 1 & \text{for all } j \in D, \\
& y_i - x_{ij} \geq 0 & \text{for all } i \in F, j \in D, \\
& \sum_{i \in F} y_i \leq k & \\
& x_{ij}, y_i \in \{0, 1\} & \text{for all } i \in F, j \in D.
\end{aligned}
$$

Lagrangean Relaxation

Take LP relaxation and move constraint $\sum y_i \leq k$ to objective function.

Let $\lambda \geq 0$ (Langrange multiplier):

$$
\begin{aligned}
\min \quad & \sum_{i \in F, j \in D} c_{ij} \cdot x_{ij} + \sum_{i \in F} \lambda \cdot y_i - \lambda \cdot k & \\
\text{s.t.} \quad & \sum_{i \in F} x_{ij} = 1 & \text{for all } j \in D, \\
& y_i - x_{ij} \geq 0 & \text{for all } i \in F, j \in D, \\
& x_{ij}, y_i \geq 0 & \text{for all } i \in F, j \in D.
\end{aligned}
$$

Dual LP:

$$
\begin{aligned}
\max \quad & \sum_{j \in D} v_j - \lambda \cdot k & \\
\text{s.t.} \quad & \sum_{j \in D} w_{ij} \leq \lambda & \text{for all } i \in F, \\
& v_j - w_{ij} \leq c_{ij} & \text{for all } i \in F, j \in D, \\
& w_{ij} \geq 0 & \text{for all } i \in F, j \in D.
\end{aligned}
$$

Algorithmic Idea

Use primal-dual algorithm for uncap. facility location with $f_i := \lambda$, $i \in F$.

This yields $S \subseteq F$ and feasible dual (v, w) with

$$\tfrac{1}{3}\sum_{j\in D}\min_{i\in S} c_{ij} + \sum_{i\in S} f_i \;\le\; \sum_{j\in D} v_j \;.$$

That is,

$$c(S) := \sum_{j\in D}\min_{i\in S} c_{ij} \;\le\; 3\Big(\sum_{j\in D} v_j - \lambda\cdot|S|\Big)\;.$$

Observation 7.12.

If the algorithm happens to output a set $S \subseteq F$ with $|S| = k$, then

$$c(S) \;\le\; 3\Big(\sum_{j\in D} v_j - \lambda\cdot k\Big) \;\le\; 3\cdot \mathrm{OPT}_k\;.$$

□

Question: Can we find $\lambda \ge 0$ such that $|S| = k$?

Choosing a Suitable $\lambda \ge 0$

Observation.

- For $\lambda = 0$ the algorithm outputs $S \subseteq F$ with $|S| \ge k$ (w.l.o.g.).
- For $\lambda = \sum_{j\in D}\sum_{i\in F} c_{ij}$, the algorithm opens a single facility only.

Idea: Use binary search until one of the following two things happen:

- find λ such that the algorithm opens exactly k facilities;
- find $\lambda_1 < \lambda_2$ with $\lambda_2 - \lambda_1 \le \frac{\varepsilon}{3} c_{\min}/|F|$; ($c_{\min} := \min\{c_{ij} \mid c_{ij} > 0\}$) for $\lambda = \lambda_\ell$ the algorithm outputs $S_\ell \subseteq F$ with $|S_1| > k > |S_2|$ and

$$c(S_\ell) \le 3\Big(\sum_{j\in D} v_j^\ell - \lambda_\ell\cdot|S_\ell|\Big)\;, \qquad \ell = 1, 2.$$

Notice that binary search terminates after $O(\log \frac{|F|\sum c_{ij}}{\varepsilon\cdot c_{\min}})$ iterations.

- In the first case, we have found a solution of cost at most $3\cdot \mathrm{OPT}_k$.
- In the second case we still need to work a little. . .

Finding a Suitable $S \subseteq F$ with $|S| \leq k$

Goal: Find $S \subseteq F$ with $|S| \leq k$ and $c(S) \leq 2(3+\varepsilon) \cdot \mathrm{OPT}_k$.

- Let $\alpha_1 := \frac{k - |S_2|}{|S_1| - |S_2|}$ and $\alpha_2 := \frac{|S_1| - k}{|S_1| - |S_2|}$.

  Notice that $\alpha_1 + \alpha_2 = 1$, $\alpha_1, \alpha_2 \geq 0$, and $\alpha_1 |S_1| + \alpha_2 |S_2| = k$.
- Let $\tilde{v} := \alpha_1 v^1 + \alpha_2 v^2$ and $\tilde{w} := \alpha_1 w^1 + \alpha_2 w^2$.

  Notice that $(\tilde{v}, \tilde{w})$ is a feasible dual solution for facility costs λ_2.

Lemma 7.13.
It holds that $\alpha_1 \cdot c(S_1) + \alpha_2 \cdot c(S_2) \leq (3+\varepsilon) \cdot \mathrm{OPT}_k$.

Proof:... □

Thus, if $\alpha_2 \geq \frac{1}{2}$, we reach our goal by setting $S := S_2$.

In the following we can thus assume that $\alpha_2 < \frac{1}{2}$...

Finding a Suitable $S \subseteq F$ with $|S| \leq k$ (cont.)

Algorithm.

1. $S := \emptyset$; for each facility $i \in S_2$, add the closest facility $h \in S_1$ to S;
2. while $|S| < k$, choose random facility in $S_1 \setminus S$ and add it to S;

Lemma 7.14.
If $\alpha_2 < \frac{1}{2}$, opening facilities as above has cost $\mathrm{E}[c(S)] \leq 2(3+\varepsilon) \cdot \mathrm{OPT}_k$.

Proof:... □

The algorithm can be derandomized by method of conditional probabilities.

Theorem 7.15.
There is a $(6+\varepsilon)$-approximation algorithm for the k-median problem. □

Theorem 7.16.
There is no 1.735-approximation algorithm unless each problem in NP has an $n^{O(\log \log n)}$ time algorithm. □