Approximation Algorithms

Lecture 5:
LP-based Approximation Algorithms for SetCover

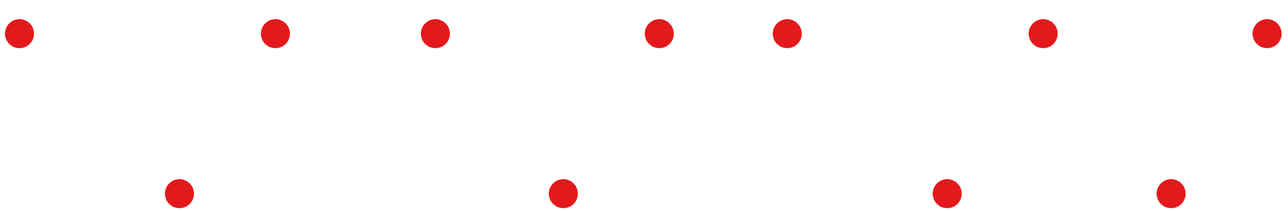
Part I:
SetCover as an ILP

Alexander Wolff

Winter 2023/24

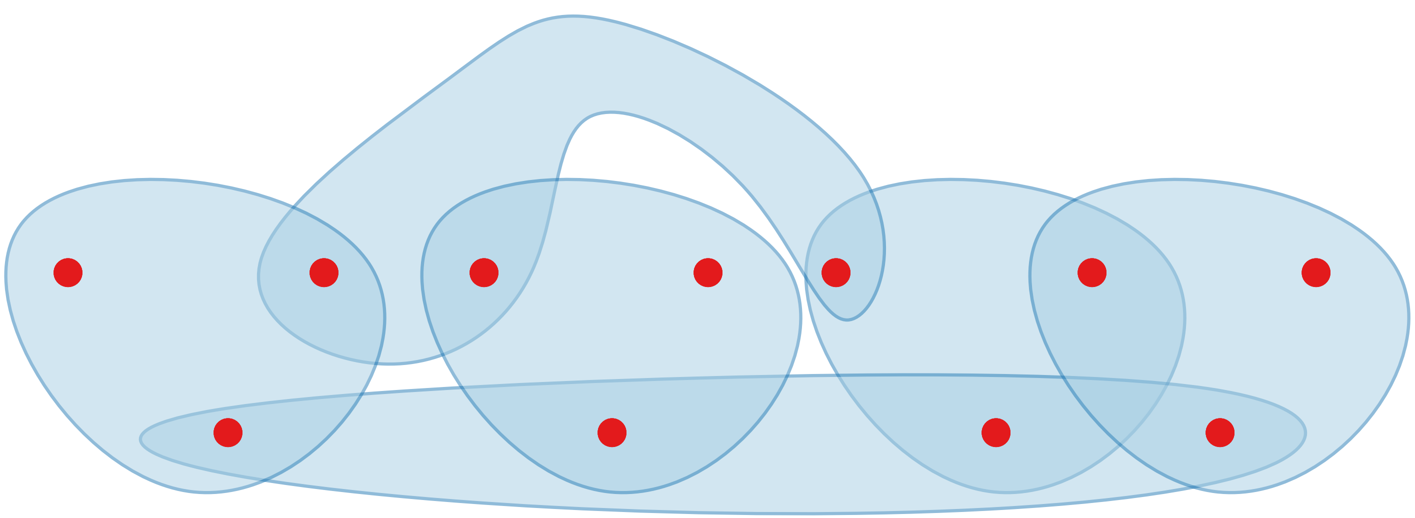
SetCover as an ILP

Ground set U



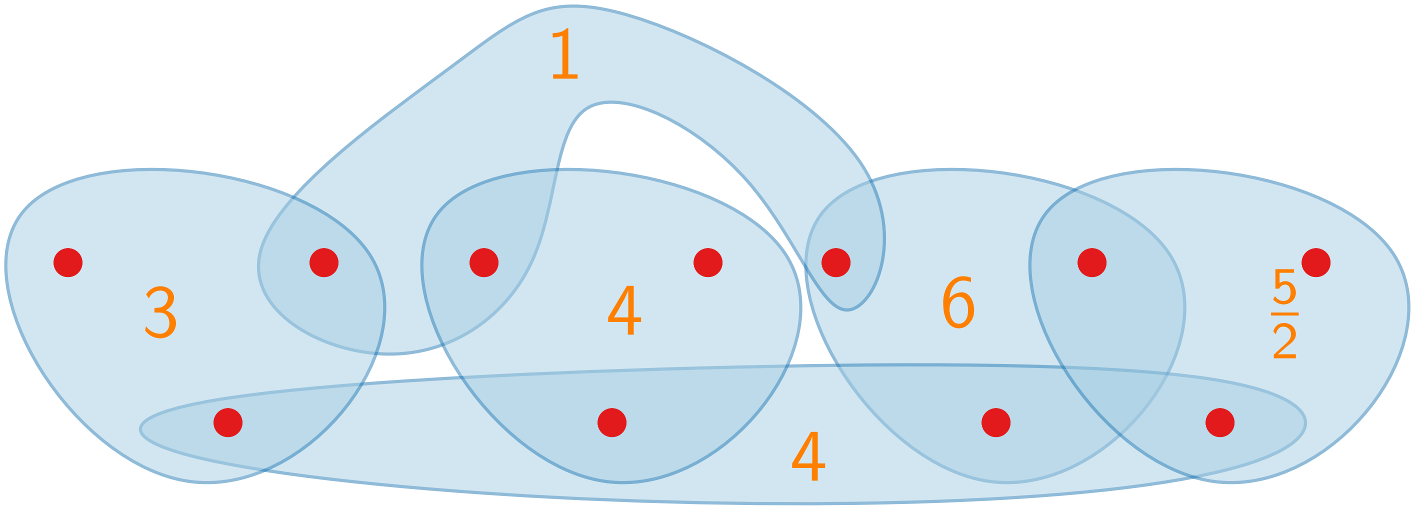
SetCover as an ILP

Ground set U
Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$



SetCover as an ILP

Ground set U
Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$
Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$

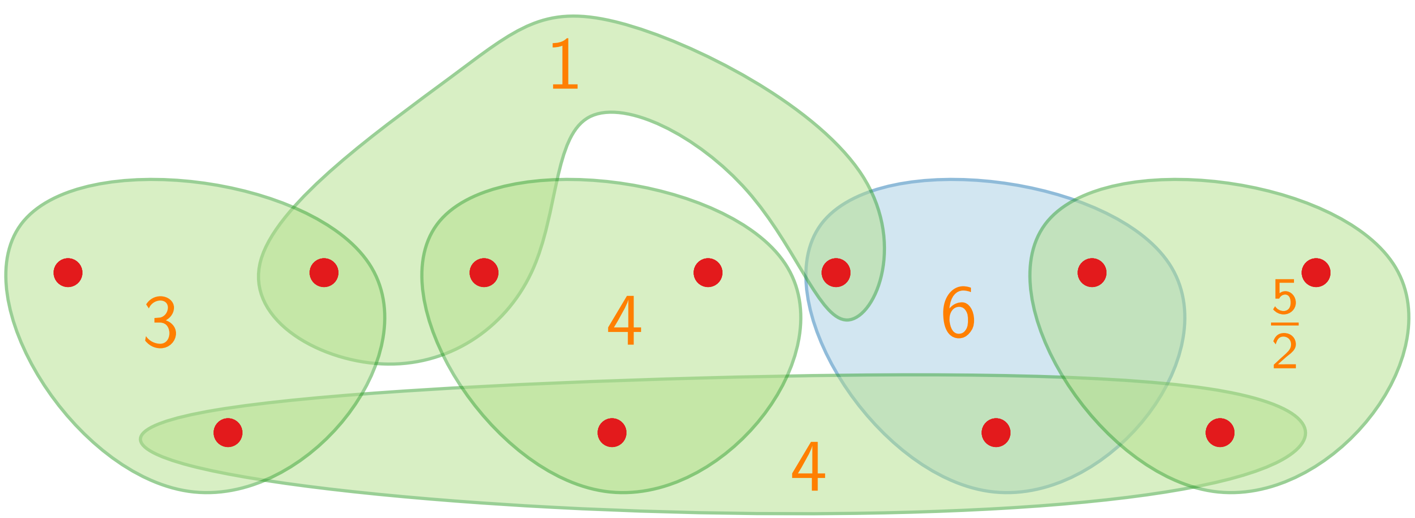


SetCover as an ILP

Ground set U

Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$

Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

SetCover as an ILP

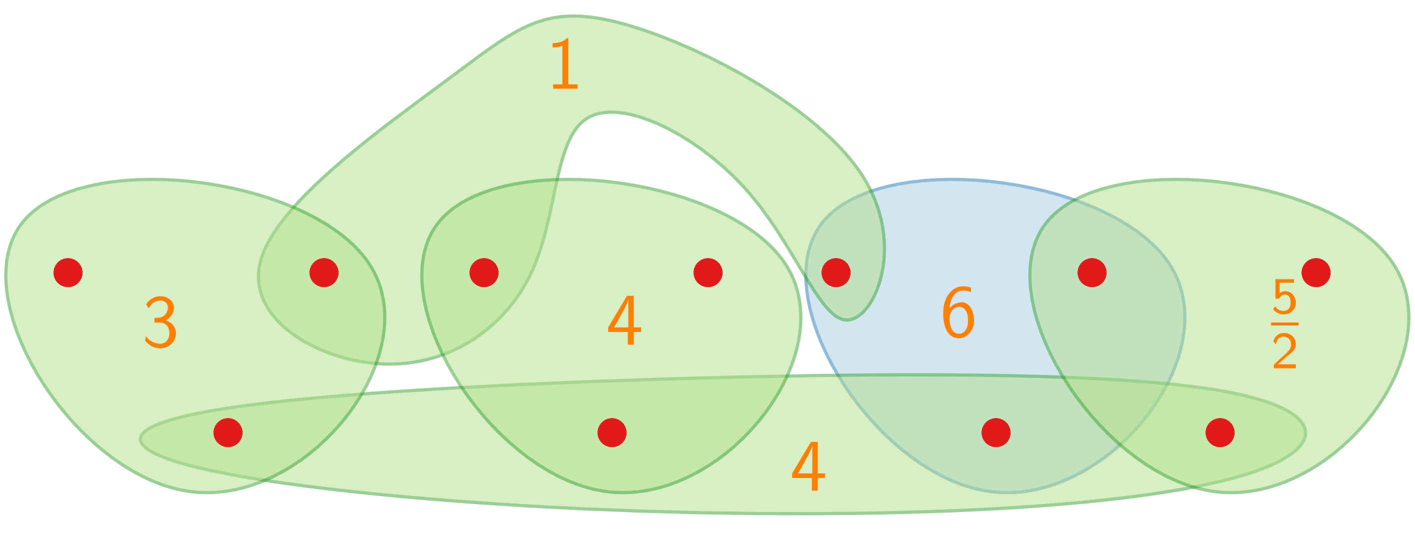
minimize

subject to

Ground set U

Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$

Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

SetCover as an ILP

minimize

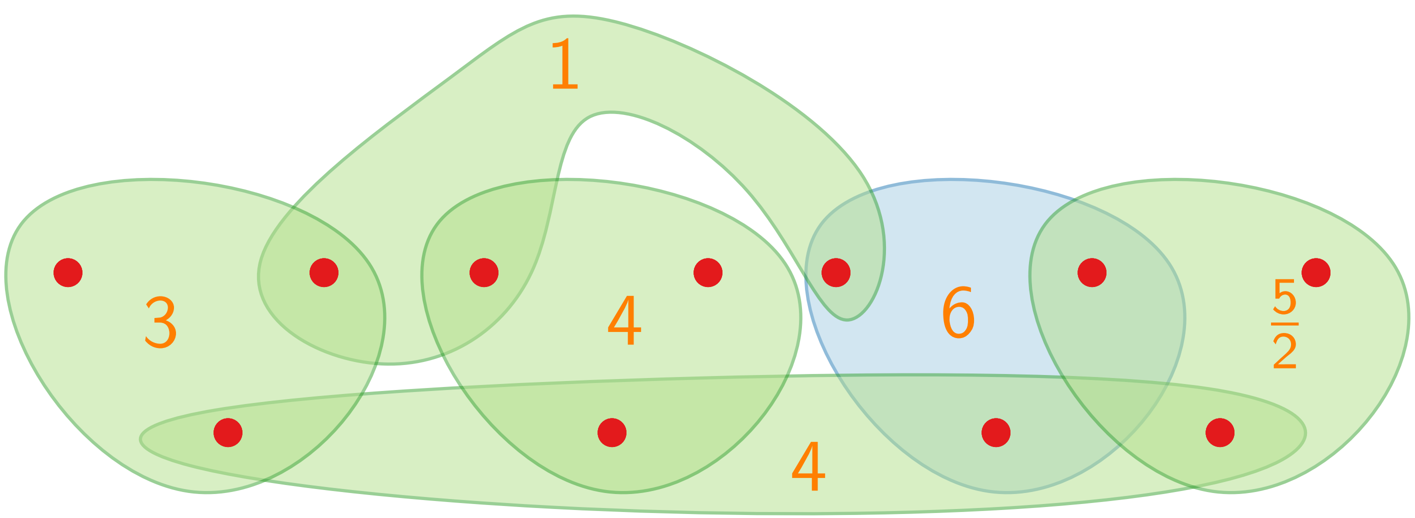
subject to

x_S $S \in \mathcal{S}$

Ground set U

Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$

Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

SetCover as an ILP

minimize

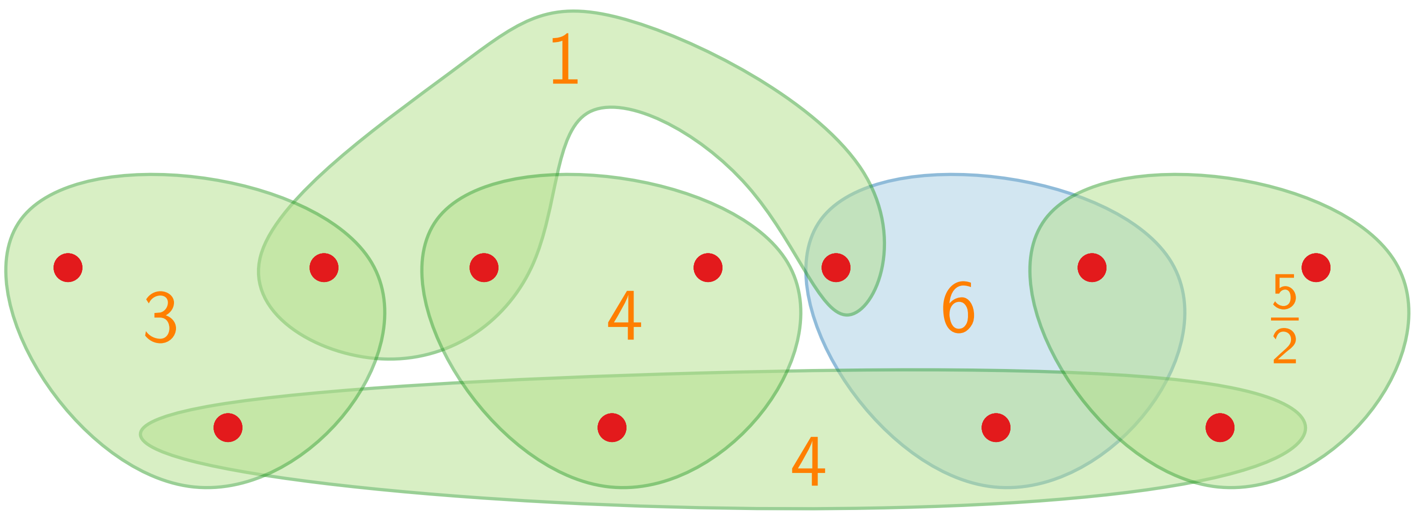
subject to

$$x_S \in \{0, 1\} \qquad S \in \mathcal{S}$$

Ground set U

Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$

Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

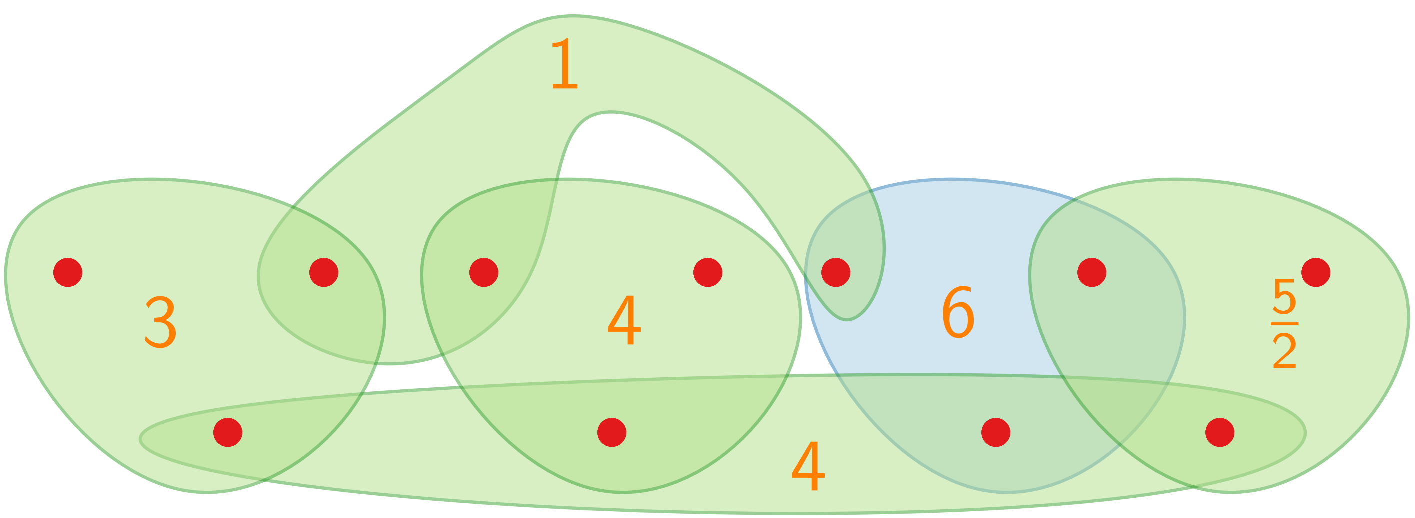
SetCover as an ILP

minimize $\sum_{S \in \mathcal{S}} c_S x_S$

subject to

$x_S \in \{0, 1\} \quad S \in \mathcal{S}$

Ground set U
Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$
Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

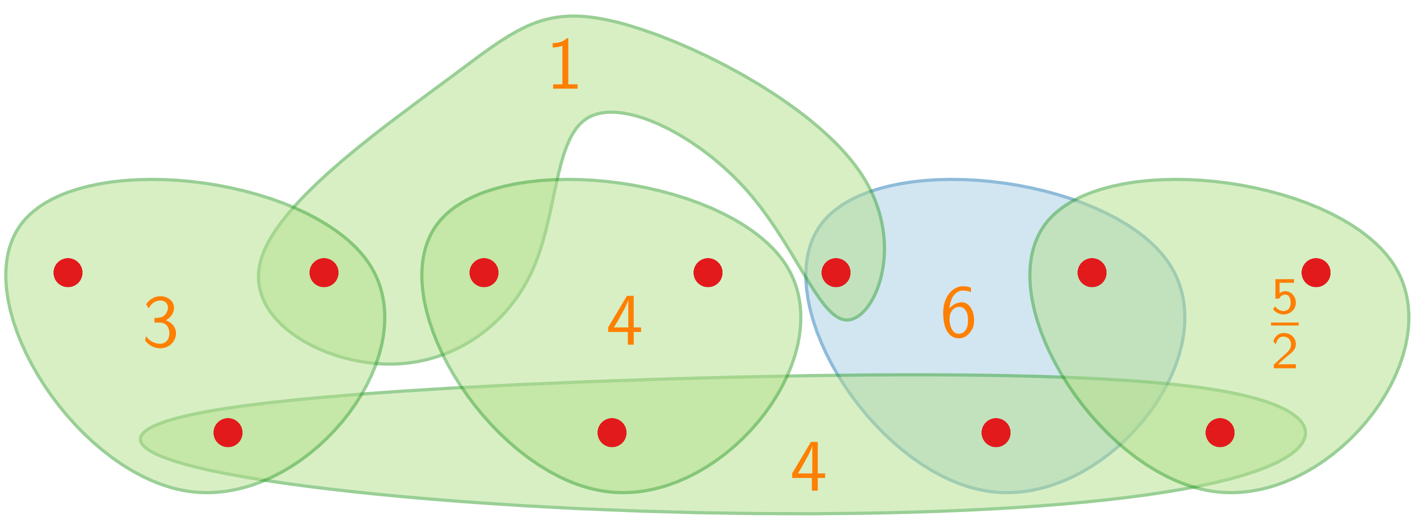
SetCover as an ILP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & & u \in U \\ & x_S \in \{0,1\} & S \in \mathcal{S} \end{array}$$

Ground set U

Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$

Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

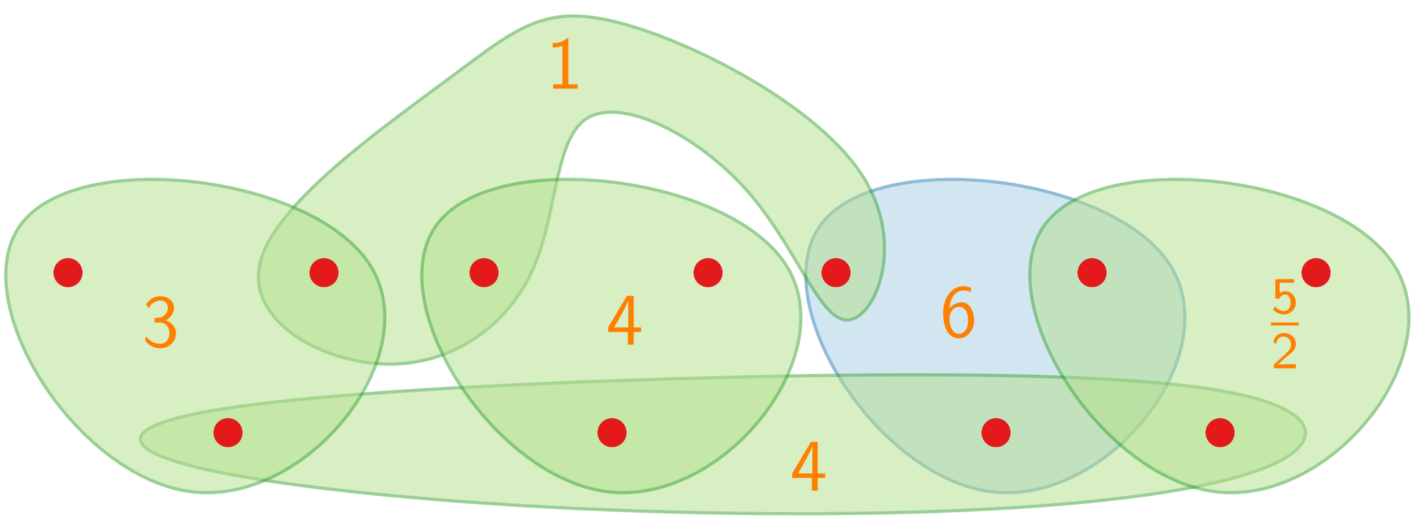
SetCover as an ILP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \in \{0,1\} & S \in \mathcal{S} \end{array}$$

Ground set U

Family $\mathcal{S} \subseteq 2^U$ with $\bigcup \mathcal{S} = U$

Costs $c \colon \mathcal{S} \to \mathbb{Q}^+$



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of U with minimum cost.

Approximation Algorithms

Lecture 5:
LP-based Approximation Algorithms for SetCover

Part II:
LP-Rounding

Technique I) LP-Rounding

Consider a minimization problem Π in ILP form.

Technique I) LP-Rounding

Consider a minimization problem Π in ILP form.

Compute a solution for the LP-relaxation.

Technique I) LP-Rounding

Consider a minimization problem Π in ILP form.

Compute a solution for the LP-relaxation.

Round to obtain an integer solution for Π.

Technique I) LP-Rounding

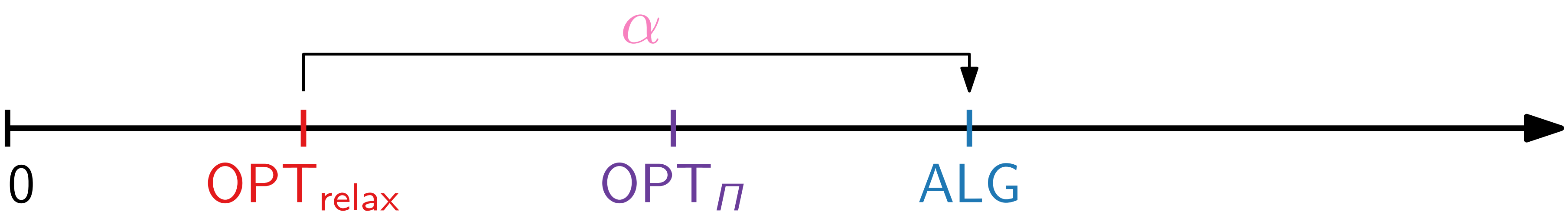
Consider a minimization problem Π in ILP form.

Compute a solution for the LP-relaxation.

Round to obtain an integer solution for Π.

Difficulty: Ensure the **feasiblity** of the solution.

Technique I) LP-Rounding



Consider a minimization problem Π in ILP form.

Compute a solution for the LP-relaxation.

Round to obtain an integer solution for Π.

Difficulty: Ensure the **feasiblity** of the solution.

Approximation factor: $\mathrm{ALG}/\mathrm{OPT}_{\Pi} \leq \mathrm{ALG}/\mathrm{OPT}_{\mathrm{relax}}$.

SetCover – LP-Relaxation

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

SetCover – LP-Relaxation

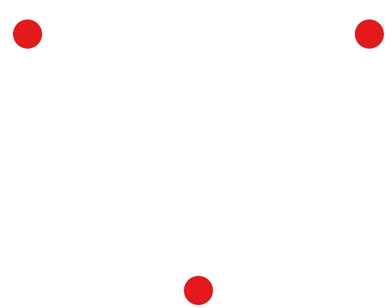
$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\
\textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\
& x_S \geq 0 & S \in \mathcal{S}
\end{array}
$$

Optimal?

SetCover – LP-Relaxation

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

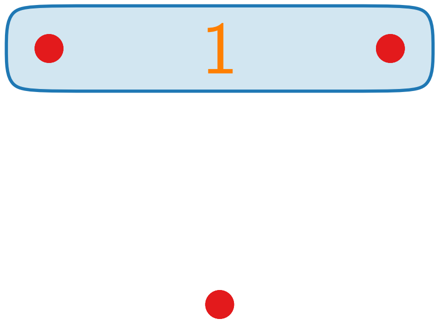
Optimal?



SetCover – LP-Relaxation

$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\
\textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\
 & x_S \geq 0 & S \in \mathcal{S}
\end{array}
$$

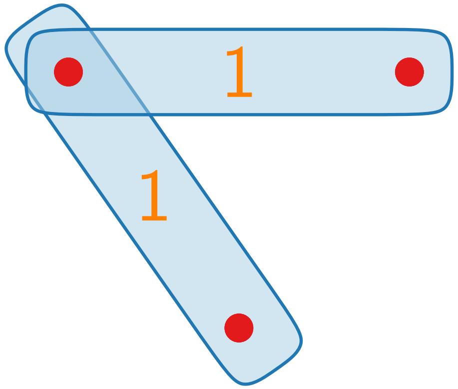
Optimal?



SetCover – LP-Relaxation

$$\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\
\textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\
 & x_S \geq 0 & S \in \mathcal{S}
\end{array}$$

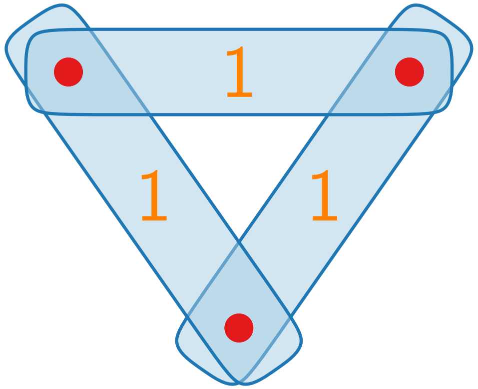
Optimal?



SetCover – LP-Relaxation

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

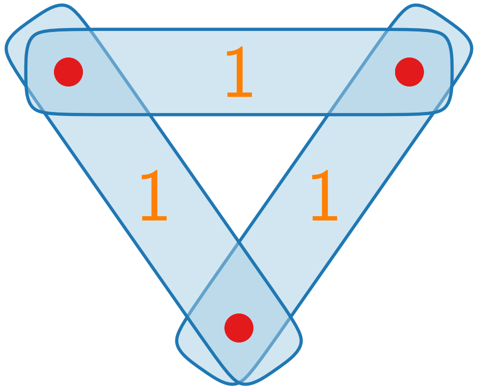
Optimal?

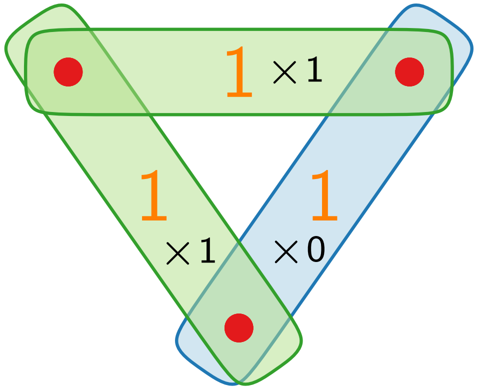


SetCover – LP-Relaxation

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

Optimal?



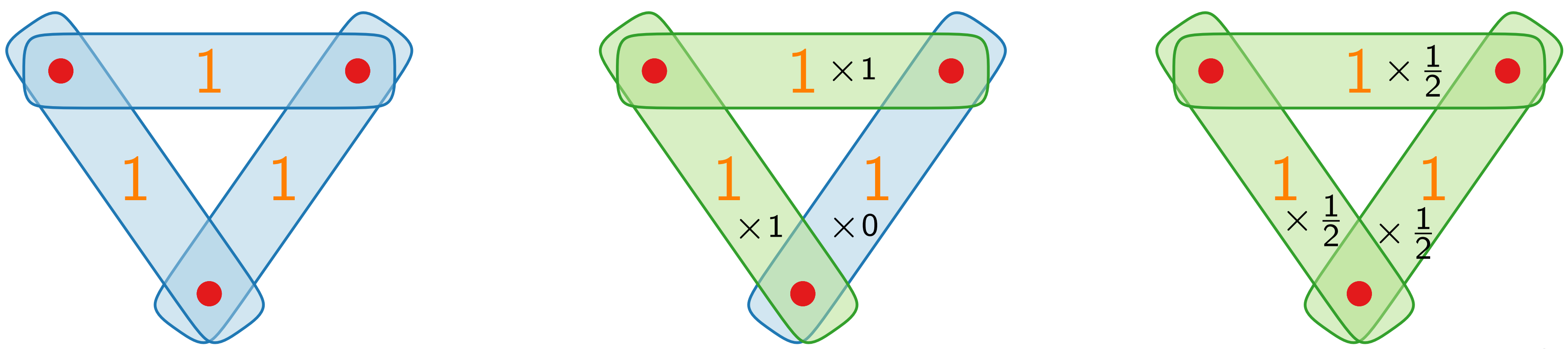


integer: 2

SetCover – LP-Relaxation

$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\
\textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\
 & x_S \geq 0 & S \in \mathcal{S}
\end{array}
$$

Optimal?



integer: 2

fractional: $\frac{3}{2}$

LP-Rounding: Approach I

$$\begin{array}{llll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & & u \in U \\ & x_S \geq 0 & & S \in \mathcal{S} \end{array}$$

LP-Rounding-One($U, \mathcal{S}, c$)

LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

LP-Rounding: Approach I

$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{aligned}$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

LP-Rounding: Approach I

$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\
\textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\
 & x_S \geq 0 & S \in \mathcal{S}
\end{array}
$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

– Scaling factor arbitrarily large.

LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

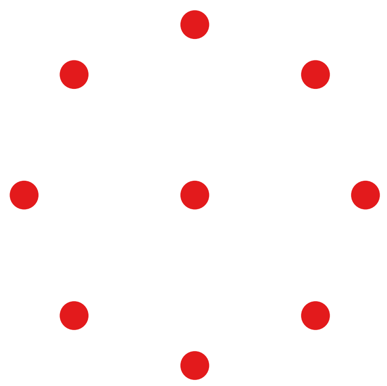
LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

– Scaling factor arbitrarily large.



LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

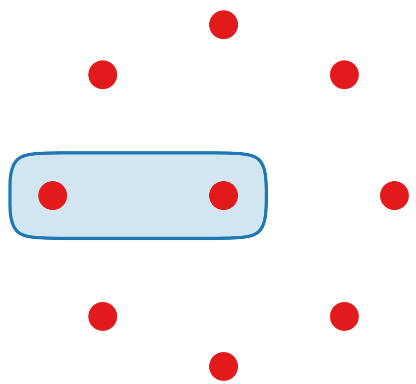
LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

– Scaling factor arbitrarily large.



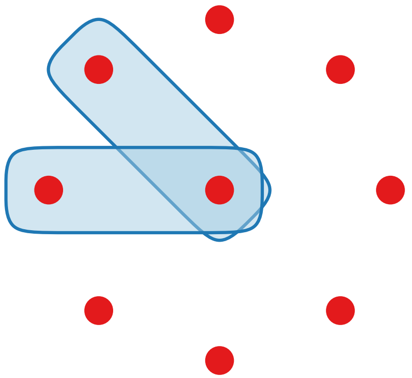
LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S\in\mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S\ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.
Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.
– Scaling factor arbitrarily large.



LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

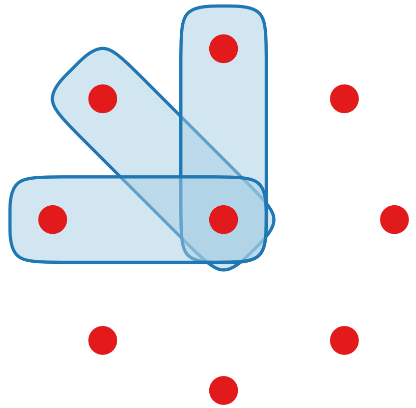
LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

– Scaling factor arbitrarily large.



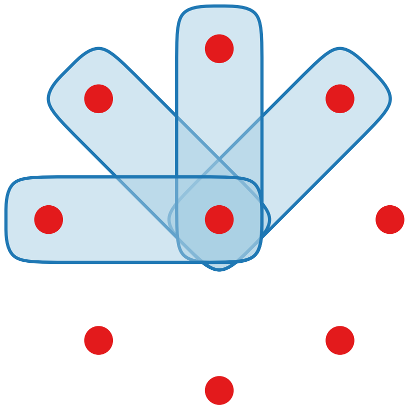
LP-Rounding: Approach I

$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad\qquad S \in \mathcal{S} \end{aligned}$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.
Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.
– Scaling factor arbitrarily large.



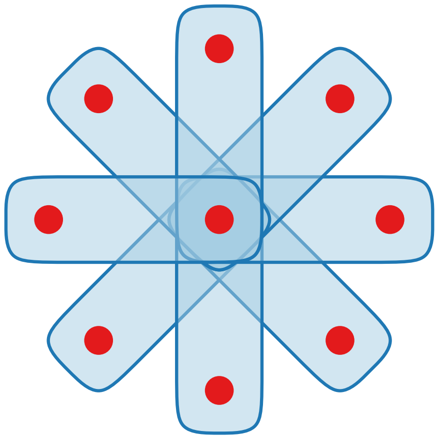
LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.
Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.
– Scaling factor arbitrarily large.



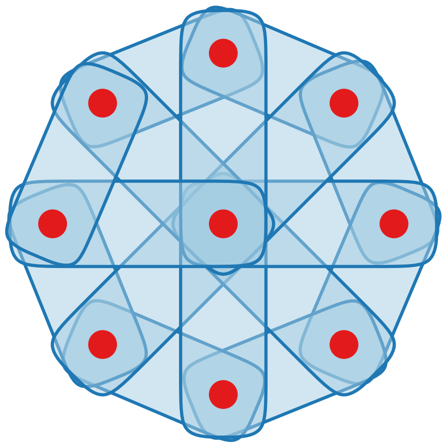
LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.
Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.
– Scaling factor arbitrarily large.



LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

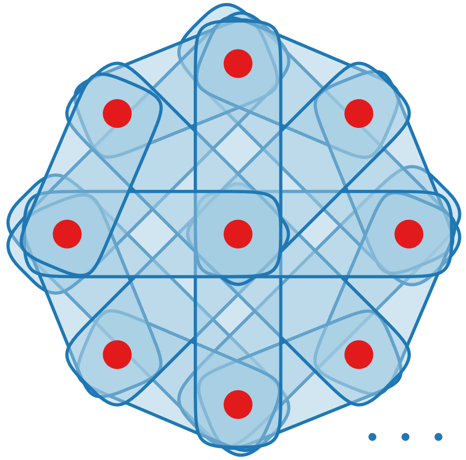
LP-Rounding-One($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

– Scaling factor arbitrarily large.



LP-Rounding: Approach I

$$\begin{array}{lll} \textbf{minimize} & \sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-One($U, \mathcal{S}, c$)

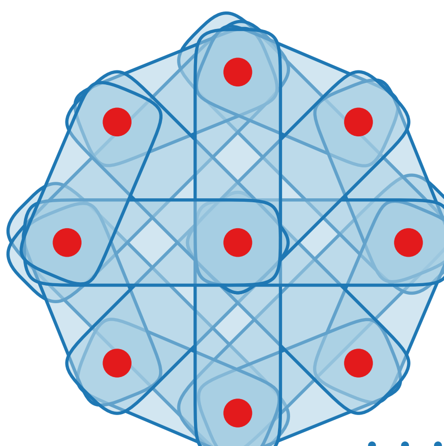
Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S > 0$ to 1.

– Generates a valid solution.

– Scaling factor arbitrarily large.

Use frequency f



LP-Rounding: Approach II

$$
\begin{array}{lll}
\textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\
\textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\
 & x_S \geq 0 & S \in \mathcal{S}
\end{array}
$$

LP-Rounding-Two($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S \geq$ to 1; remaining to 0.

Let f be the frequency of (i.e., the number of sets containing) the most frequent element.

LP-Rounding: Approach II

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-Two($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.

Round each x_S with $x_S \geq 1/f$ to 1; remaining to 0.

Let f be the frequency of (i.e., the number of sets containing) the most frequent element.

LP-Rounding: Approach II

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

LP-Rounding-Two($U, \mathcal{S}, c$)

Compute optimal solution x for LP-relaxation.
Round each x_S with $x_S \geq 1/f$ to 1; remaining to 0.

Let f be the frequency of (i.e., the number of sets containing) the most frequent element.

Theorem. LP-Rounding-Two is a factor-f approximation algorithm for SetCover.

Approximation Algorithms

Lecture 5:
LP-based Approximation Algorithms for SetCover

Part III:
The Primal-Dual Schema

Technique II) Primal–Dual Approach

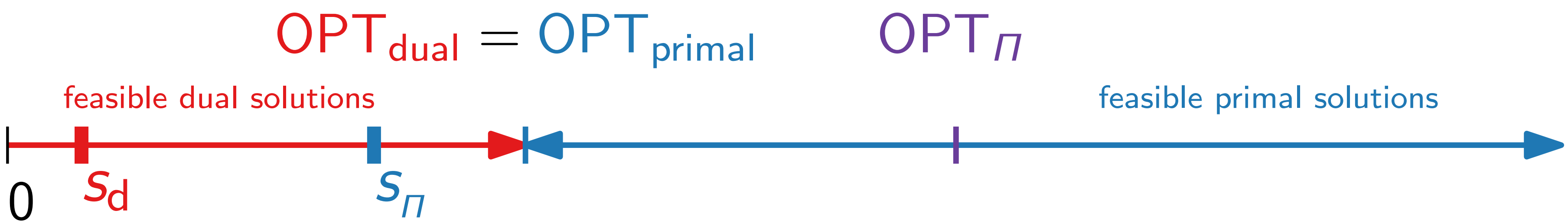
Consider a minimization problem Π in ILP form.

Technique II) Primal–Dual Approach

Consider a minimization problem Π in ILP form.

- Start with (trivial) feasible dual solution and infeasible primal solution (e.g., all variables $= 0$).

Technique II) Primal–Dual Approach



Consider a minimization problem Π in ILP form.

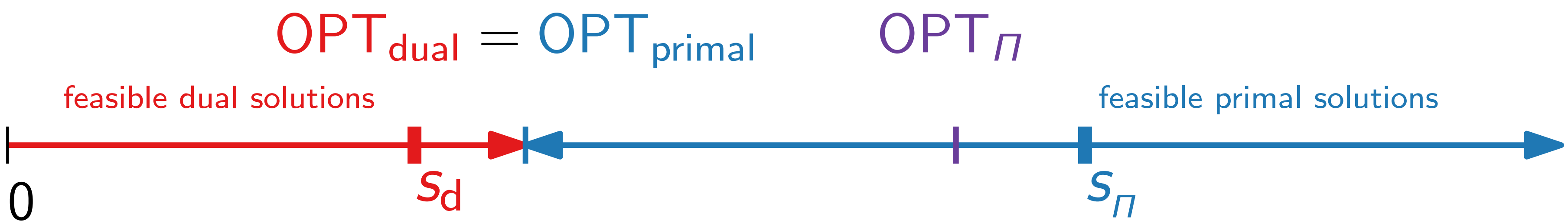
- Start with (trivial) feasible dual solution and infeasible primal solution (e.g., all variables $= 0$).

Technique II) Primal–Dual Approach

Consider a minimization problem Π in ILP form.

- Start with (trivial) feasible dual solution and infeasible primal solution (e.g., all variables $= 0$).
- Compute dual solution s_{d} and integral primal solution s_{Π} for Π iteratively:
  Increase s_{d} according to CS and make s_{Π} "more feasible".

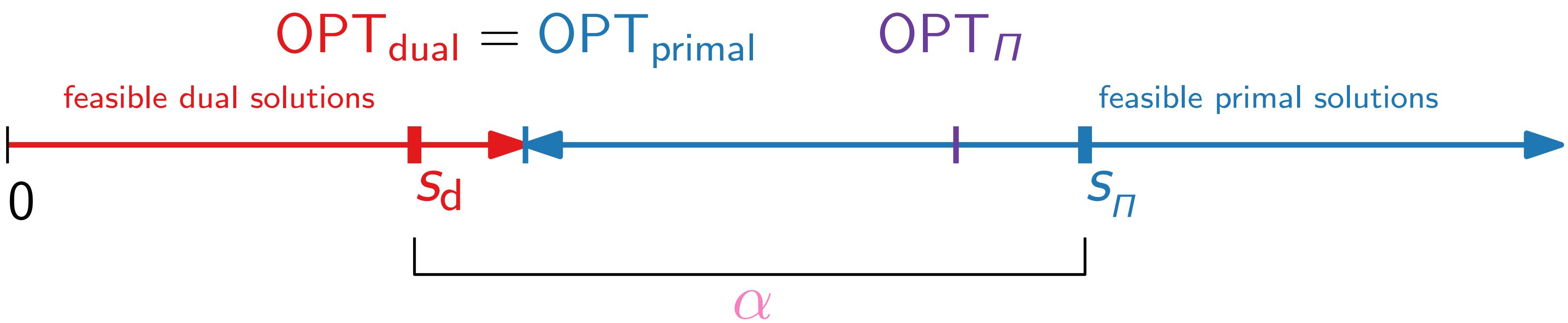
Technique II) Primal–Dual Approach



Consider a minimization problem Π in ILP form.

- Start with (trivial) feasible dual solution and infeasible primal solution (e.g., all variables $= 0$).
- Compute dual solution s_{d} and integral primal solution s_{Π} for Π iteratively:
  Increase s_{d} according to CS and make s_{Π} "more feasible".

Technique II) Primal–Dual Approach

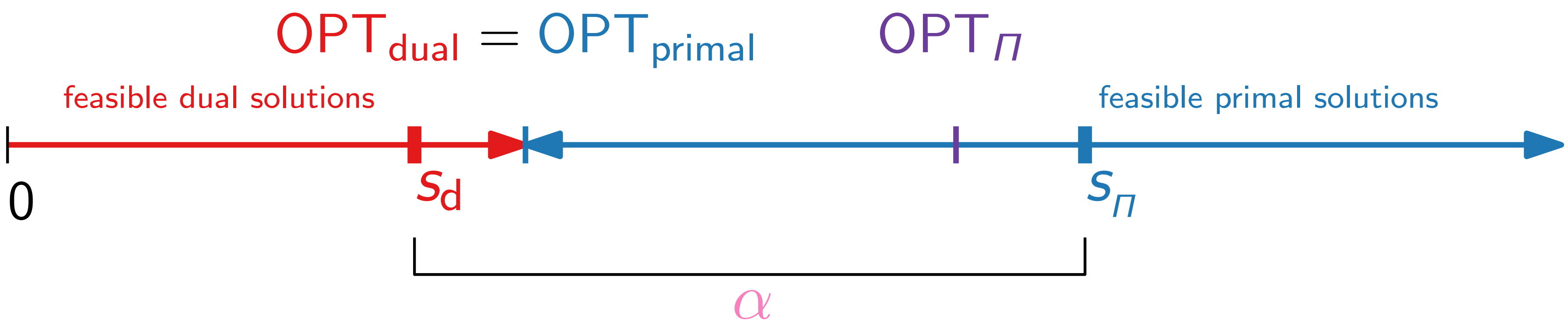


Consider a minimization problem Π in ILP form.

- Start with (trivial) feasible dual solution and infeasible primal solution (e.g., all variables $= 0$).
- Compute dual solution s_{d} and integral primal solution s_{Π} for Π iteratively:
  Increase s_{d} according to CS and make s_{Π} "more feasible".

Approximation factor $\leq \mathrm{obj}(s_{\Pi})/\mathrm{obj}(s_{\mathrm{d}})$

Technique II) Primal–Dual Approach



Consider a minimization problem Π in ILP form.

- Start with (trivial) feasible dual solution and infeasible primal solution (e.g., all variables $= 0$).
- Compute dual solution s_{d} and integral primal solution s_{Π} for Π iteratively:
  Increase s_{d} according to CS and make s_{Π} "more feasible".

Approximation factor $\leq \mathrm{obj}(s_{\Pi})/\mathrm{obj}(s_{\mathrm{d}})$

Advantage: Don't need LP-"machinery"; possibly faster, more flexible.

SetCover – Dual LP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

SetCover – Dual LP

minimize $\sum_{S \in \mathcal{S}} c_S x_S$

subject to $\sum_{S \ni u} x_S \geq 1 \qquad u \in U$

$x_S \geq 0 \qquad S \in \mathcal{S}$

maximize

subject to

SetCover – Dual LP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & & \\ \textbf{subject to} & & \\ & y_u \geq 0 & u \in U \end{array}$$

SetCover – Dual LP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & & \\ & y_u \geq 0 & u \in U \end{array}$$

SetCover – Dual LP

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

Complementary Slackness

$$\begin{array}{llcl} \textbf{minimize} & c^\intercal x & & \\ \textbf{subject to} & Ax & \geq & b \\ & x & \geq & 0 \end{array}$$

$$\begin{array}{llcl} \textbf{maximize} & b^\intercal y & & \\ \textbf{subject to} & A^\intercal y & \leq & c \\ & y & \geq & 0 \end{array}$$

Theorem. Let $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_m)$ be valid solutions for the primal and dual program (resp.). Then x and y are optimal if and only if the following conditions are met:

Primal CS:
For each $j = 1, \ldots, n$: $\quad x_j = 0 \;$ or $\; \sum_{i=1}^m a_{ij} y_i = c_j$

Dual CS:
For each $i = 1, \ldots, m$: $\quad y_i = 0 \;$ or $\; \sum_{j=1}^n a_{ij} x_j = b_i$

Relaxing Complementary Slackness

$$\begin{array}{llll} \textbf{minimize} & c^\intercal x & & \\ \textbf{subject to} & Ax & \geq & b \\ & x & \geq & 0 \end{array}$$

$$\begin{array}{llll} \textbf{maximize} & b^\intercal y & & \\ \textbf{subject to} & A^\intercal y & \leq & c \\ & y & \geq & 0 \end{array}$$

Primal CS:

For each $j = 1, \ldots, n$: $\quad x_j = 0 \;$ or $\; \sum_{i=1}^m a_{ij} y_i = c_j$

Dual CS:

For each $i = 1, \ldots, m$: $\quad y_i = 0 \;$ or $\; \sum_{j=1}^n a_{ij} x_j = b_i$

$$\Leftrightarrow \sum_{j=1}^{n} c_j x_j = \sum_{i=1}^{m} b_i y_i$$

Relaxing Complementary Slackness

$$\begin{array}{llll} \textbf{minimize} & c^\intercal x & & \\ \textbf{subject to} & Ax & \geq & b \\ & x & \geq & 0 \end{array}$$

$$\begin{array}{llll} \textbf{maximize} & b^\intercal y & & \\ \textbf{subject to} & A^\intercal y & \leq & c \\ & y & \geq & 0 \end{array}$$

~~**Primal CS**~~: Relaxed Primal CS

For each $j = 1, \ldots, n$: $x_j = 0$ or $\cancel{\sum_{i=1}^m a_{ij} y_i = c_j}$

$$c_j/\alpha \leq \sum_{i=1}^m a_{ij} y_i \leq c_j$$

Dual CS:

For each $i = 1, \ldots, m$: $y_i = 0$ or $\sum_{j=1}^n a_{ij} x_j = b_i$

$$\Leftrightarrow \sum_{j=1}^{n} c_j x_j = \sum_{i=1}^{m} b_i y_i$$

Relaxing Complementary Slackness

minimize	$c^\intercal x$		
subject to	Ax	$\geq$	b
	x	$\geq$	0

maximize	$b^\intercal y$		
subject to	$A^\intercal y$	$\leq$	c
	y	$\geq$	0

~~**Primal CS**~~: Relaxed Primal CS

For each $j = 1, \ldots, n$: $x_j = 0$ or $\cancel{\sum_{i=1}^m a_{ij} y_i = c_j}$

$$c_j / \alpha \leq \sum_{i=1}^m a_{ij} y_i \leq c_j$$

~~**Dual CS**~~: Relaxed Dual CS

For each $i = 1, \ldots, m$: $y_i = 0$ or $\cancel{\sum_{j=1}^n a_{ij} x_j = b_i}$

$$b_i \leq \sum_{j=1}^n a_{ij} x_j \leq \beta \cdot b_i$$

$$\Leftrightarrow \sum_{j=1}^{n} c_j x_j = \sum_{i=1}^{m} b_i y_i$$

Relaxing Complementary Slackness

minimize	$c^\intercal x$		
subject to	Ax	$\geq$	b
	x	$\geq$	0

maximize	$b^\intercal y$		
subject to	$A^\intercal y$	$\leq$	c
	y	$\geq$	0

~~**Primal CS**~~: Relaxed Primal CS

For each $j = 1, \ldots, n$: $x_j = 0$ or $\cancel{\sum_{i=1}^m a_{ij} y_i = c_j}$

$$c_j/\alpha \leq \sum_{i=1}^m a_{ij} y_i \leq c_j$$

~~**Dual CS**~~: Relaxed Dual CS

For each $i = 1, \ldots, m$: $y_i = 0$ or $\cancel{\sum_{j=1}^n a_{ij} x_j = b_i}$

$$b_i \leq \sum_{j=1}^n a_{ij} x_j \leq \beta \cdot b_i$$

$$\cancel{\Leftrightarrow \sum_{j=1}^n c_j x_j = \sum_{i=1}^m b_i y_i} \quad \Rightarrow \sum_{j=1}^n c_j x_j \leq \alpha\beta \sum_{i=1}^m b_i y_i \leq \alpha\beta \cdot \mathrm{OPT}_{\mathrm{LP}}$$

Primal–Dual Schema

Start with a feasible dual and infeasible primal solution (often trivial).

Primal–Dual Schema

Start with a feasible dual and infeasible primal solution (often trivial).

“Improve” the feasibility of the primal solution...

Primal–Dual Schema

Start with a feasible dual and infeasible primal solution (often trivial).

"Improve" the feasibility of the primal solution...

...and simultaneously the objective value of the dual solution.

Primal–Dual Schema

Start with a feasible dual and infeasible primal solution (often trivial).

"Improve" the feasibility of the primal solution...

...and simultaneously the objective value of the dual solution.

Do so until the relaxed CS conditions are met.

Primal–Dual Schema

Start with a feasible dual and infeasible primal solution (often trivial).

“Improve” the feasibility of the primal solution…

…and simultaneously the objective value of the dual solution.

Do so until the relaxed CS conditions are met.

Maintain that the primal solution is integer-valued.

Primal–Dual Schema

Start with a feasible dual and infeasible primal solution (often trivial).

"Improve" the feasibility of the primal solution...

...and simultaneously the objective value of the dual solution.

Do so until the relaxed CS conditions are met.

Maintain that the primal solution is integer-valued.

The feasibility of the primal solution and the relaxed CS conditions provide an approximation ratio.

Relaxed CS for SetCover

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

Relaxed CS for SetCover

$$\begin{array}{lll} \textbf{minimize} & \sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & \sum_{u \in U} y_u & \\ \textbf{subject to} & \sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

(Unrelaxed) primal CS:

Relaxed CS for SetCover

$$\begin{array}{lll} \textbf{minimize} & \displaystyle\sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} & \displaystyle\sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{array}$$

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow$

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

only chooses critical sets

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

only chooses critical sets

Relaxed dual CS:

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

only chooses critical sets

Relaxed dual CS: $y_u \neq 0 \Rightarrow$

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

only chooses critical sets

Relaxed dual CS: $y_u \neq 0 \Rightarrow 1 \leq \sum_{S \ni u} x_S \leq f$

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

only chooses critical sets

Relaxed dual CS: $y_u \neq 0 \Rightarrow 1 \leq \sum_{S \ni u} x_S \leq f \cdot 1$

Relaxed CS for SetCover

minimize	$\sum_{S \in \mathcal{S}} c_S x_S$	
subject to	$\sum_{S \ni u} x_S \geq 1$	$u \in U$
	$x_S \geq 0$	$S \in \mathcal{S}$

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

critical set

(Unrelaxed) primal CS: $x_S \neq 0 \Rightarrow \sum_{u \in S} y_u = c_S$

only chooses critical sets

trivial for binary x

Relaxed dual CS: $y_u \neq 0 \Rightarrow 1 \leq \sum_{S \ni u} x_S \leq f \cdot 1$

Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$

repeat

until all elements are covered.

return x

Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$
repeat
    Select an uncovered element u.
until all elements are covered.
return x

Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$
repeat
  Select an uncovered element u.
  Increase y_u until a set S is critical $(\sum_{u' \in S} y_{u'} = c_S)$.
until all elements are covered.
return x

Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$

repeat

  Select an uncovered element u.

  Increase y_u until a set S is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  Select all critical sets and update x.

until all elements are covered.

return x

Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

Increase y_u until a set S is critical $\left(\sum_{u' \in S} y_{u'} = c_S\right)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

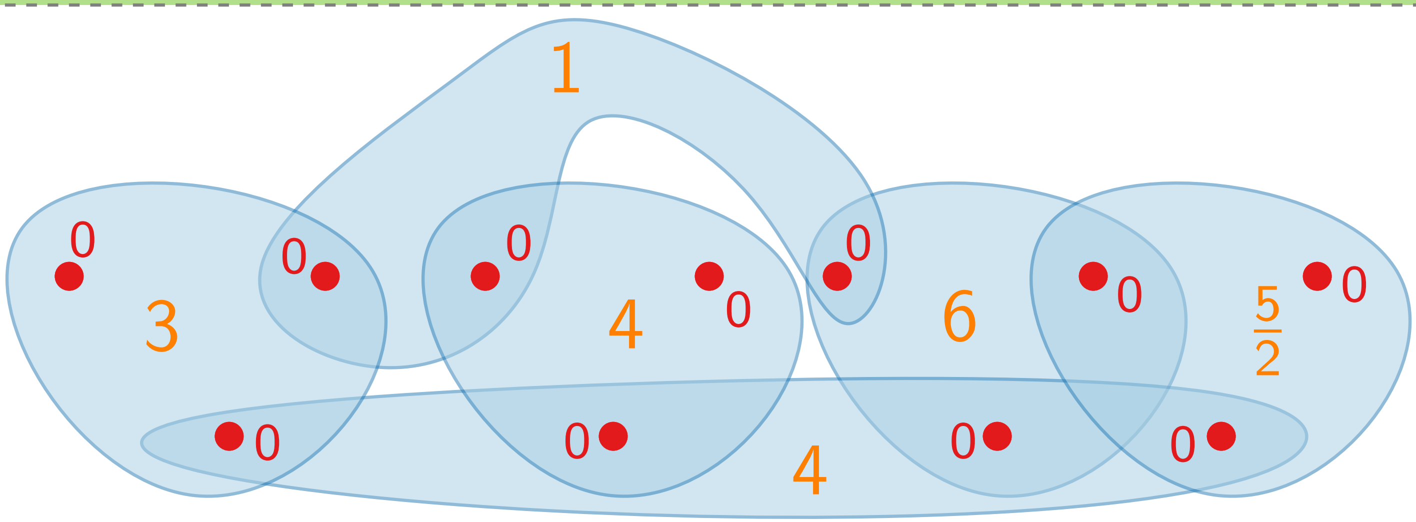
until all elements are covered.

return x

Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

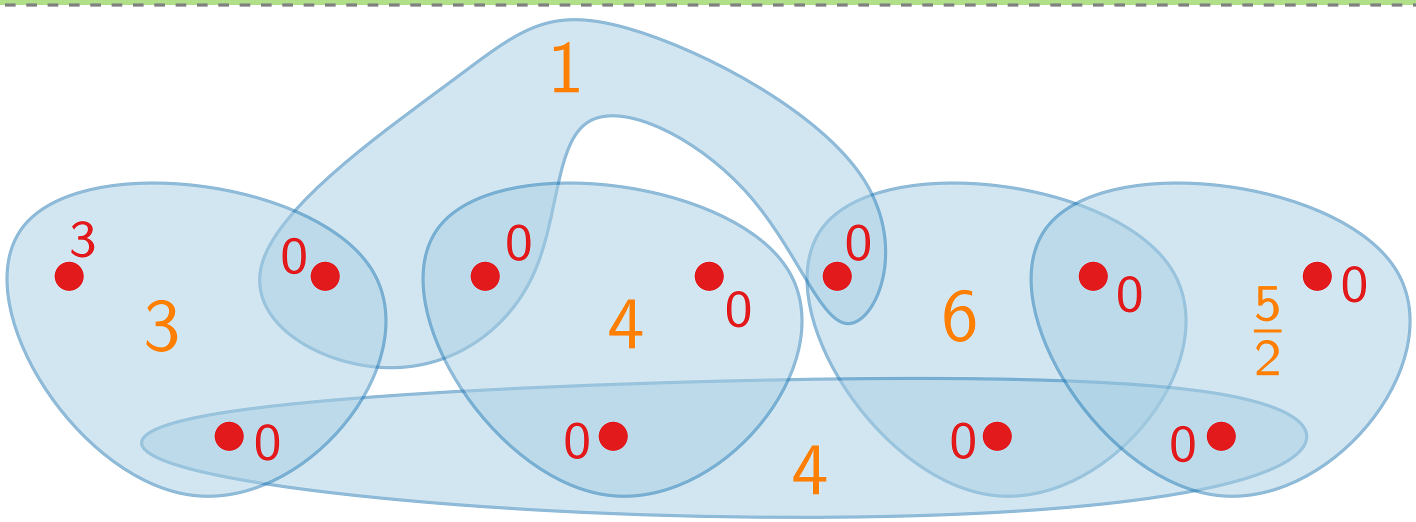
$x \leftarrow 0$, $y \leftarrow 0$
repeat
  Select an uncovered element u.
  Increase y_u until a set S is critical ($\sum_{u' \in S} y_{u'} = c_S$).
  Select all critical sets and update x.
  Mark all elements in these sets as covered.
until all elements are covered.
return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$
repeat
  Select an uncovered element u.
  Increase y_u until a set S is critical $\left(\sum_{u' \in S} y_{u'} = c_S\right)$.
  Select all critical sets and update x.
  Mark all elements in these sets as covered.
until all elements are covered.
return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

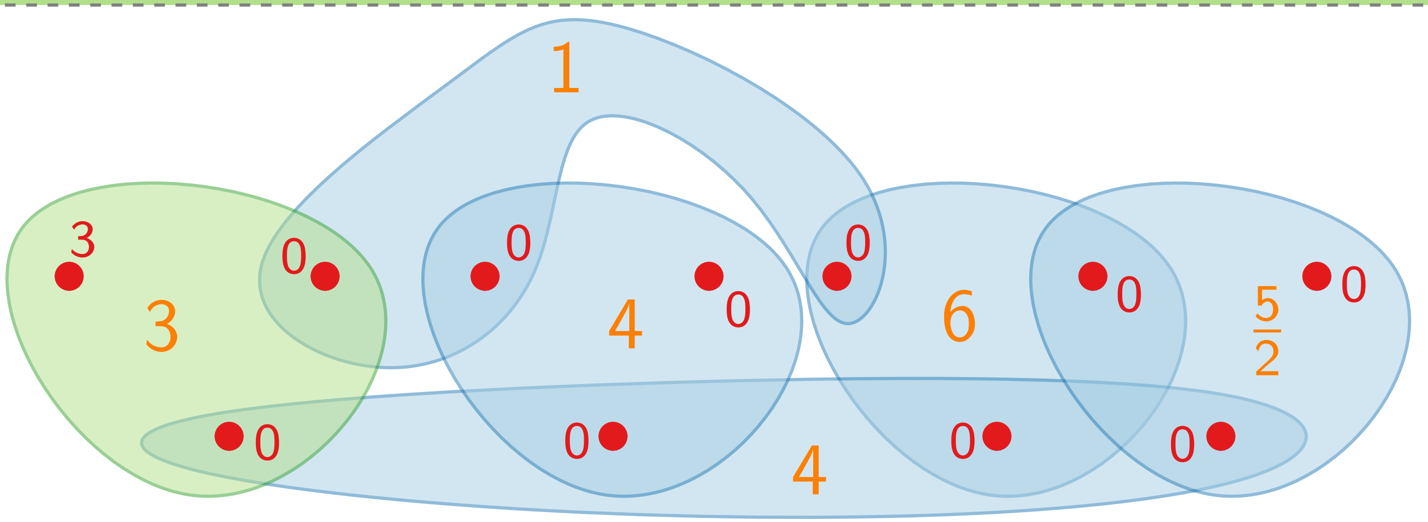
Increase y_u until a set S is critical ($\sum_{u' \in S} y_{u'} = c_S$).

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

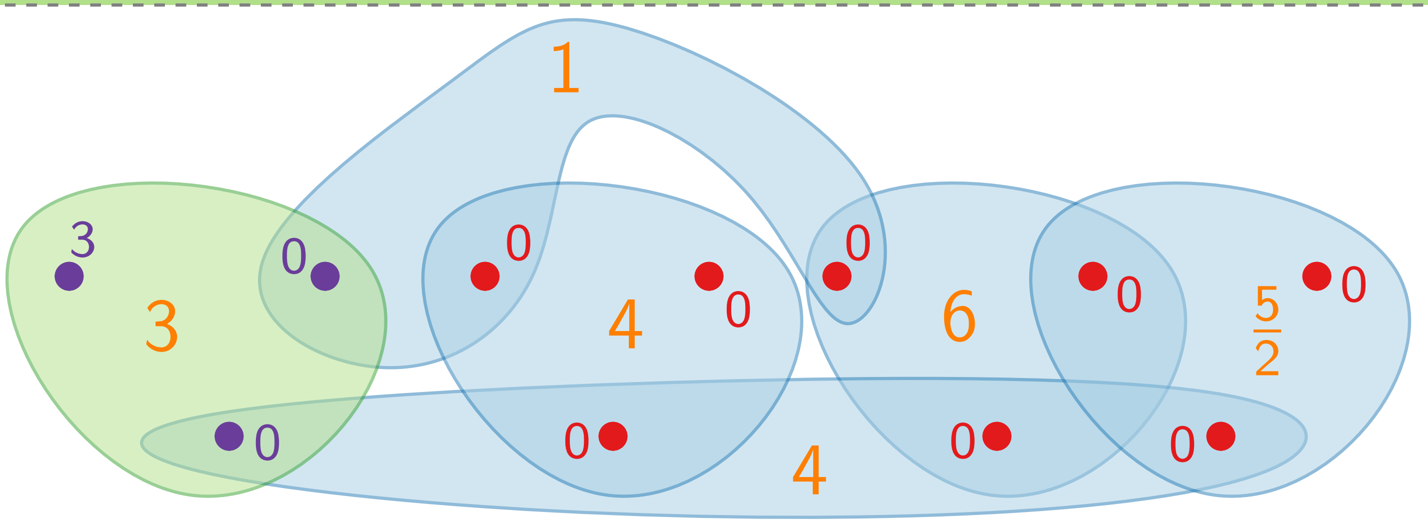
Increase y_u until a set S is critical ($\sum_{u' \in S} y_{u'} = c_S$).

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

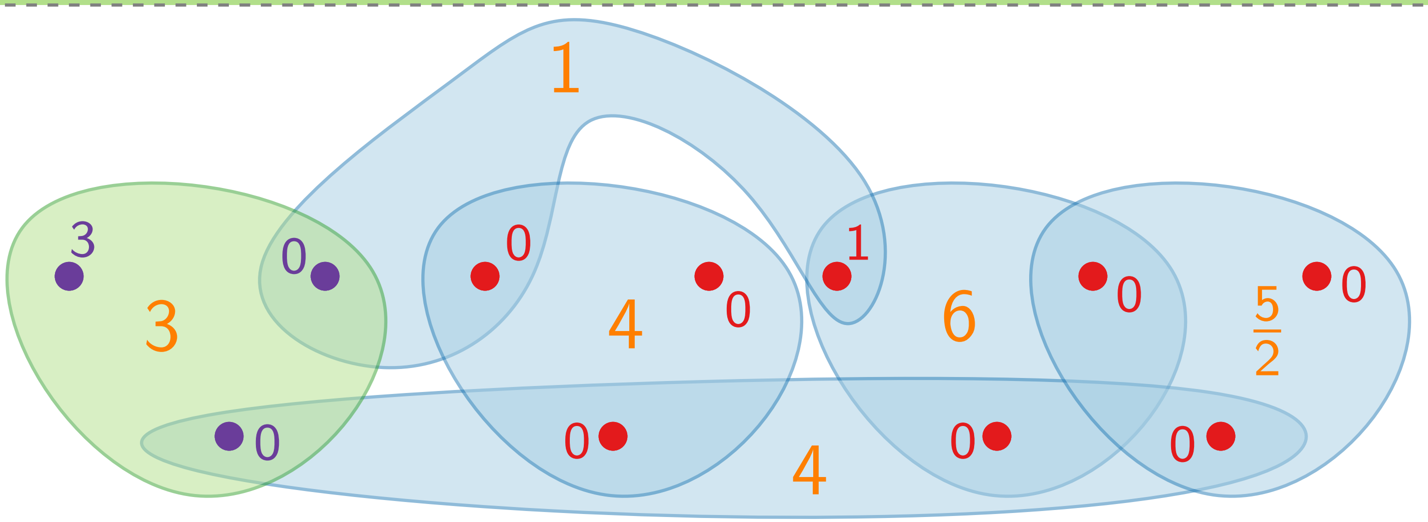
Increase y_u until a set S is critical $(\sum_{u' \in S} y_{u'} = c_S)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

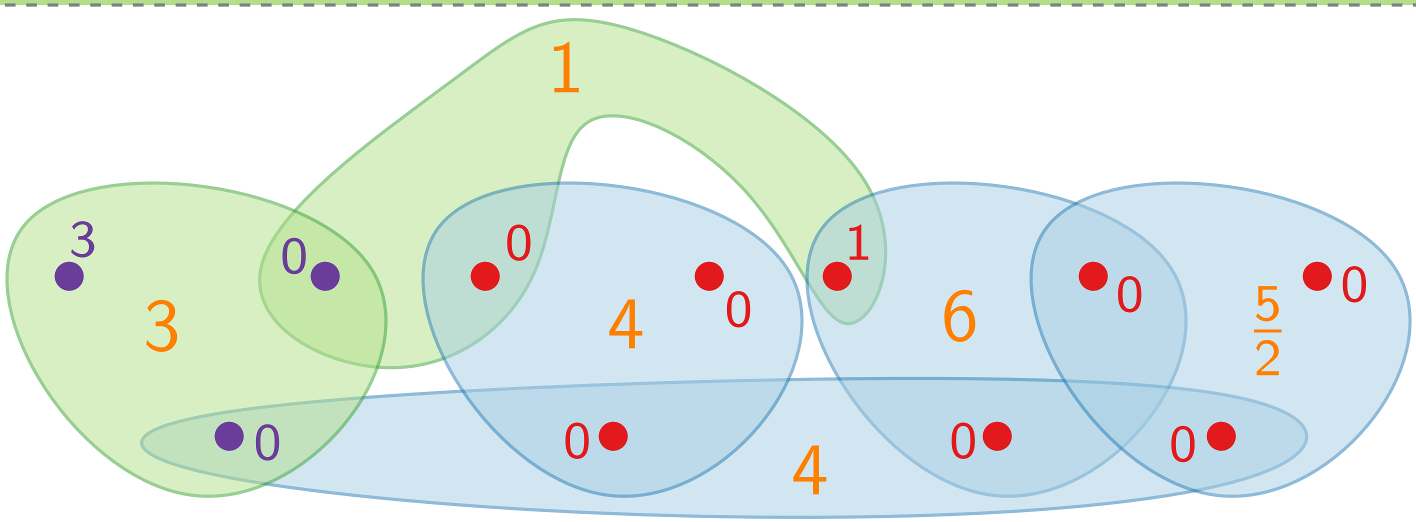
Increase y_u until a set S is critical $(\sum_{u' \in S} y_{u'} = c_S)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

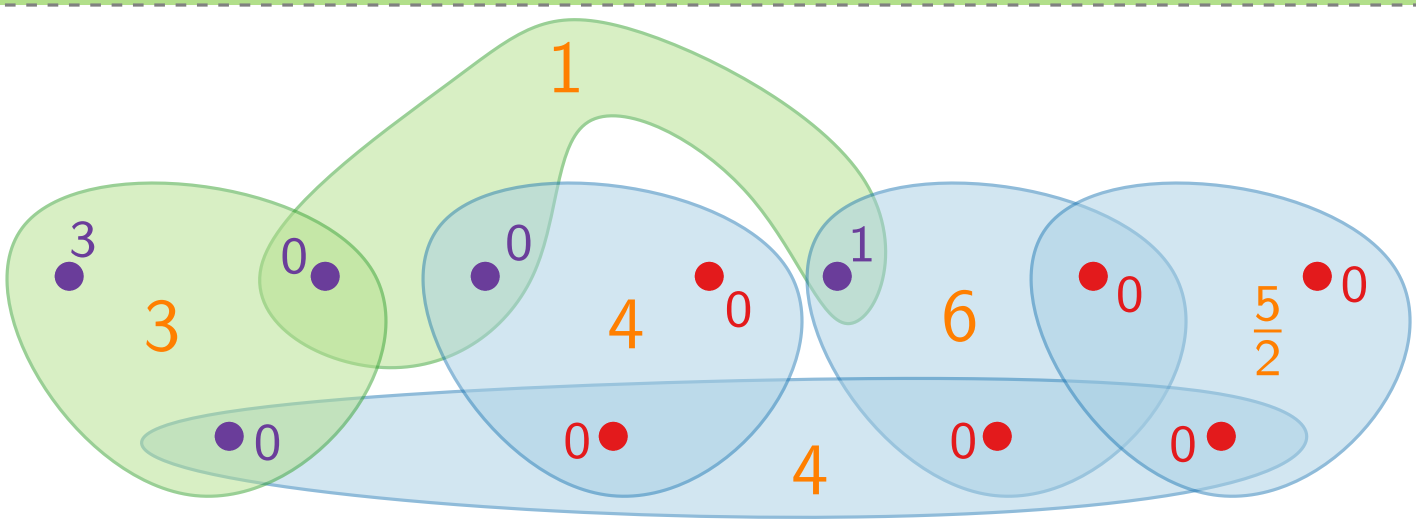
Increase y_u until a set S is critical ($\sum_{u' \in S} y_{u'} = c_S$).

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

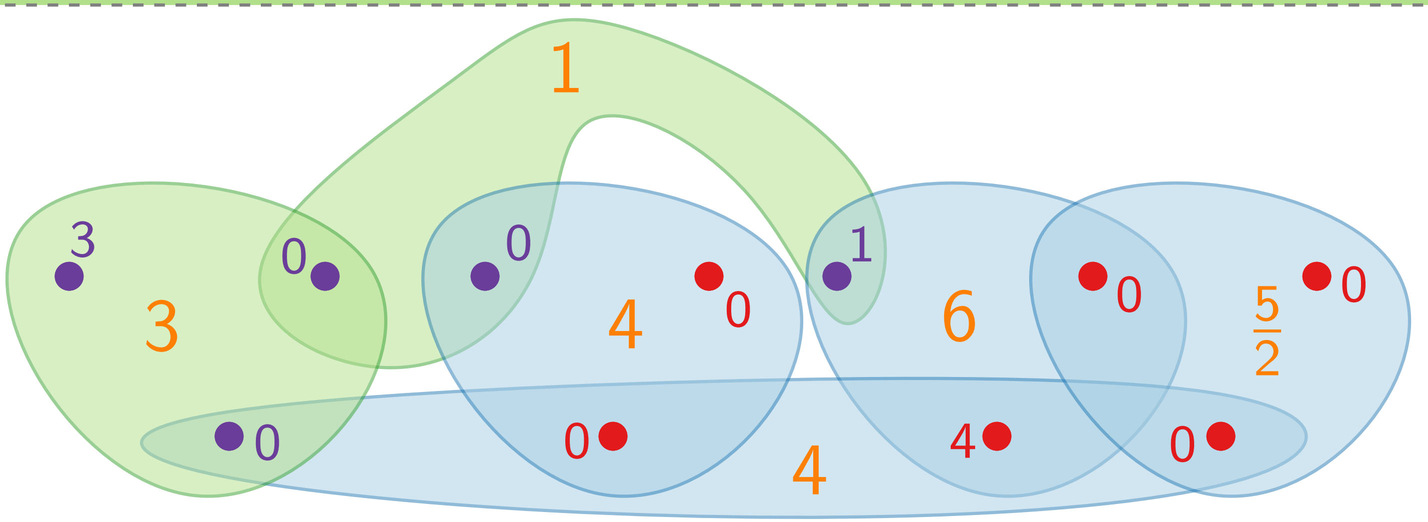
return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$
repeat
  Select an uncovered element u.
  Increase y_u until a set S is critical ($\sum_{u' \in S} y_{u'} = c_S$).
  Select all critical sets and update x.
  Mark all elements in these sets as covered.
until all elements are covered.
return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$

repeat

Select an uncovered element u.

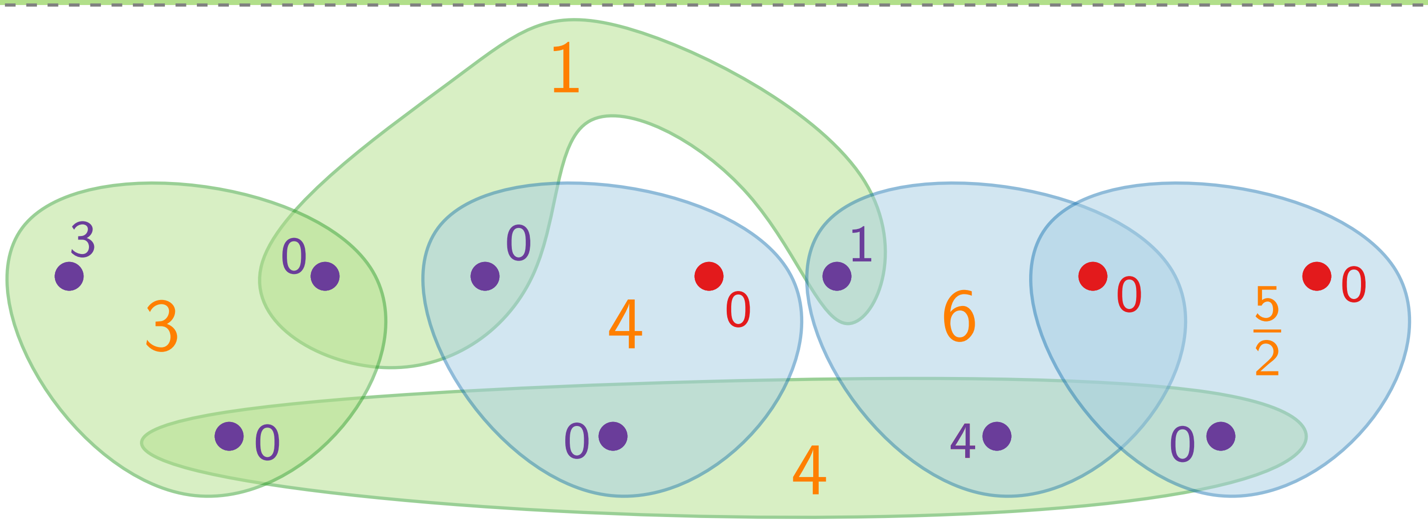
Increase y_u until a set S is critical $(\sum_{u' \in S} y_{u'} = c_S)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

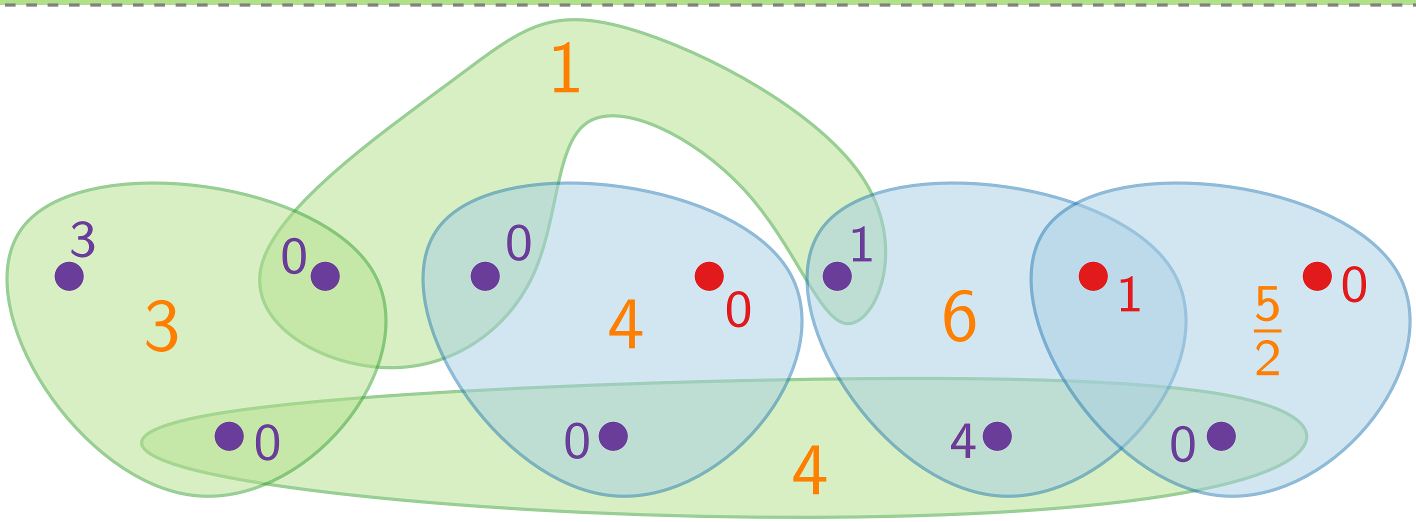
Increase y_u until a set S is critical $\left(\sum_{u' \in S} y_{u'} = c_S\right)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

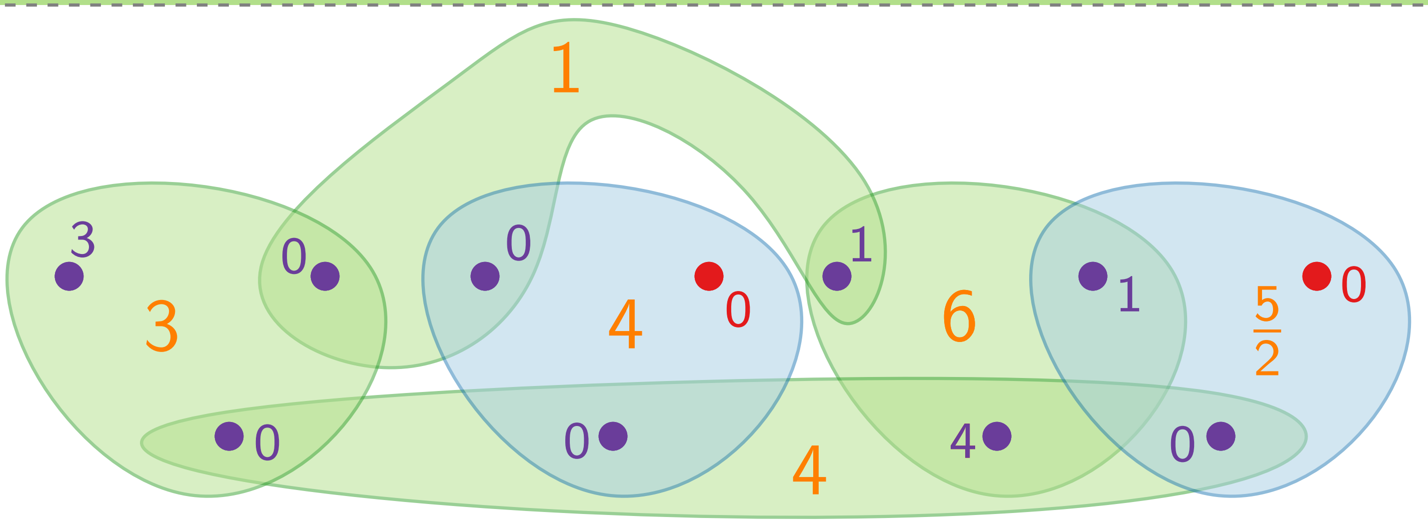
return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$
repeat
Select an uncovered element u.
Increase y_u until a set S is critical $(\sum_{u' \in S} y_{u'} = c_S)$.
Select all critical sets and update x.
Mark all elements in these sets as covered.
until all elements are covered.
return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0$, $y \leftarrow 0$

repeat

Select an uncovered element u.

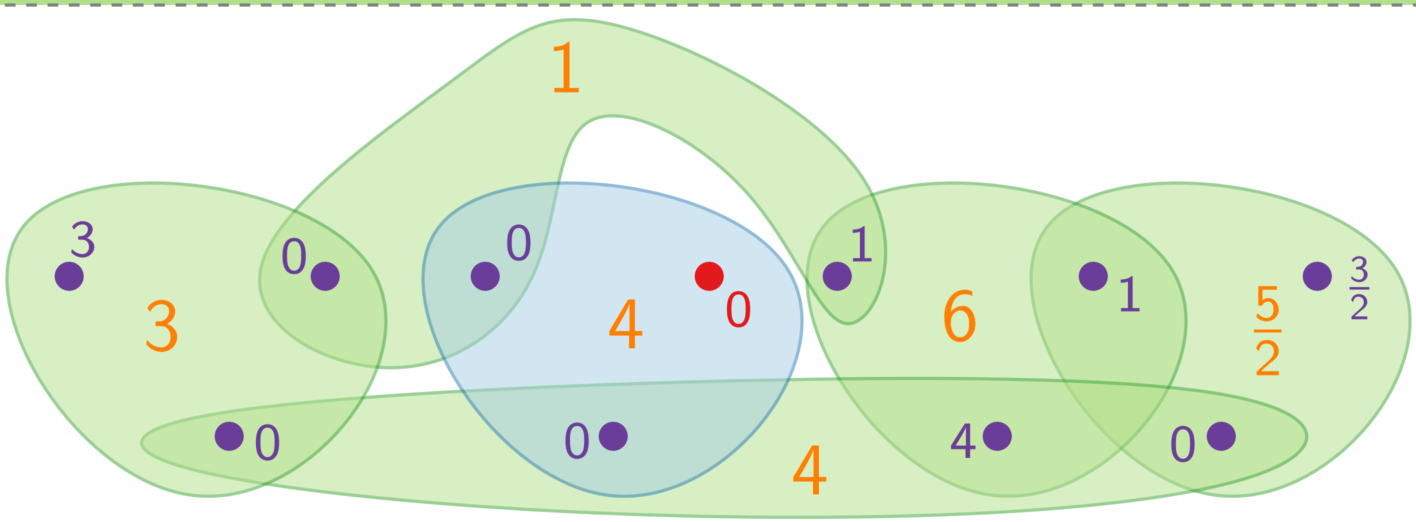
Increase y_u until a set S is critical $\left(\sum_{u' \in S} y_{u'} = c_S\right)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

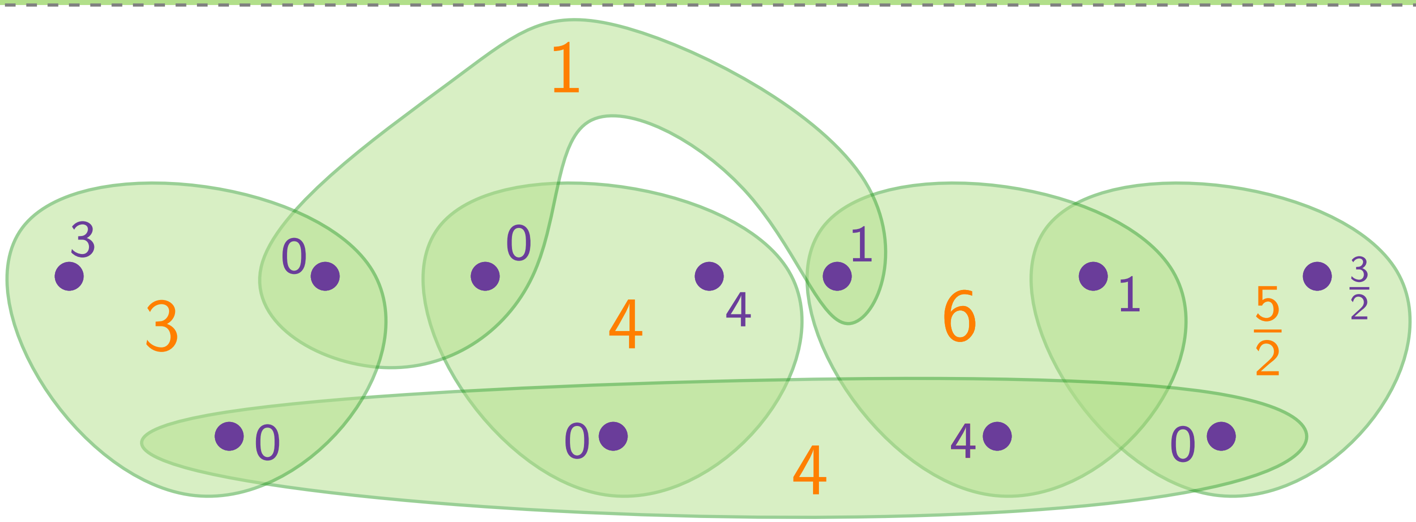
Increase y_u until a set S is critical $\left(\sum_{u' \in S} y_{u'} = c_S\right)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

return x



Primal–Dual Schema for SetCover

PrimalDualSetCover($U, \mathcal{S}, c$)

$x \leftarrow 0, y \leftarrow 0$

repeat

Select an uncovered element u.

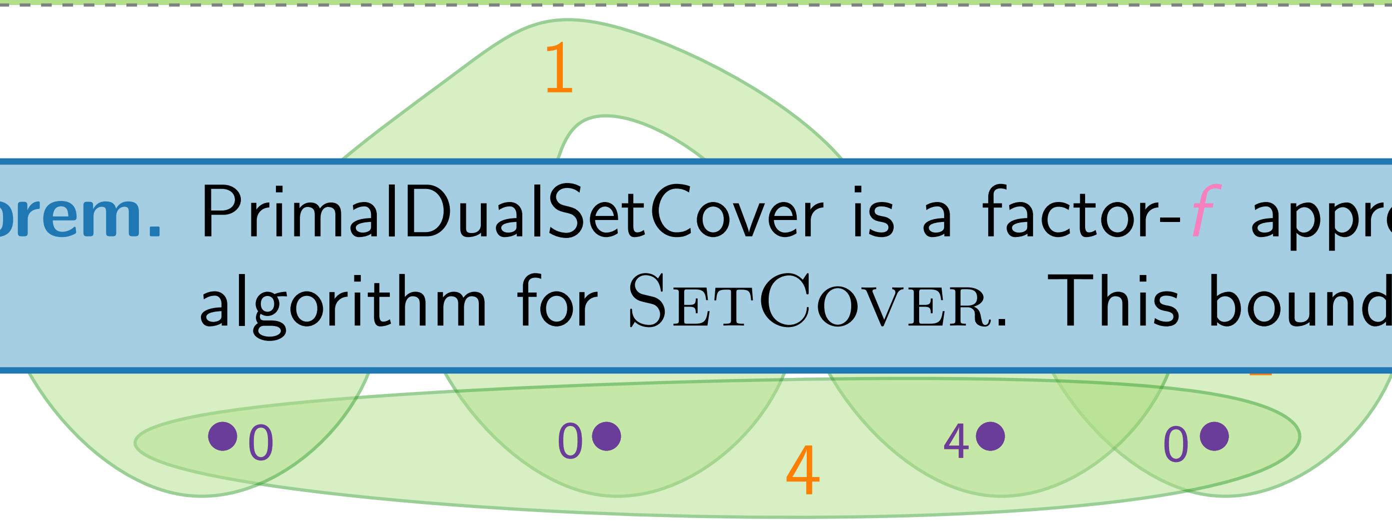
Increase y_u until a set S is critical $\left(\sum_{u' \in S} y_{u'} = c_S\right)$.

Select all critical sets and update x.

Mark all elements in these sets as covered.

until all elements are covered.

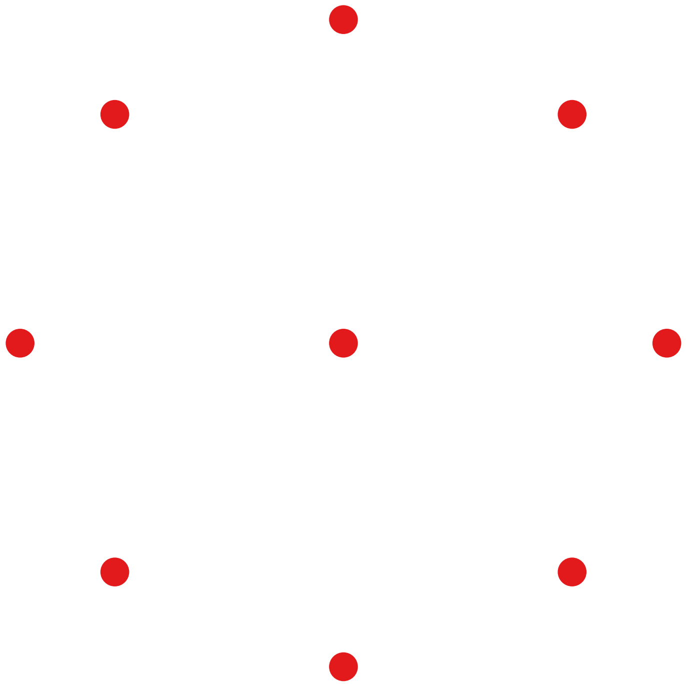
return x



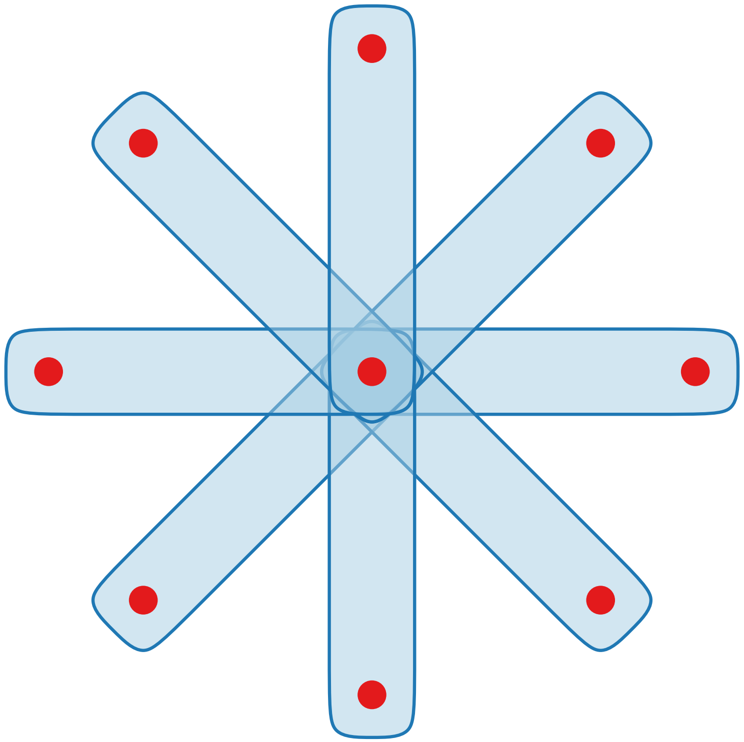
Theorem. PrimalDualSetCover is a factor-f approximation algorithm for SetCover. This bound is tight.

Tight Example

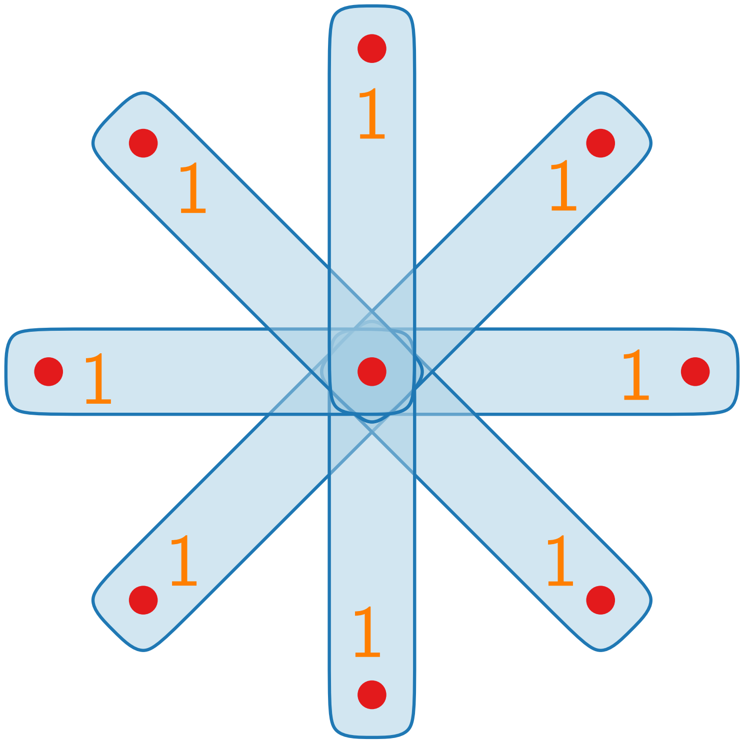
Tight Example



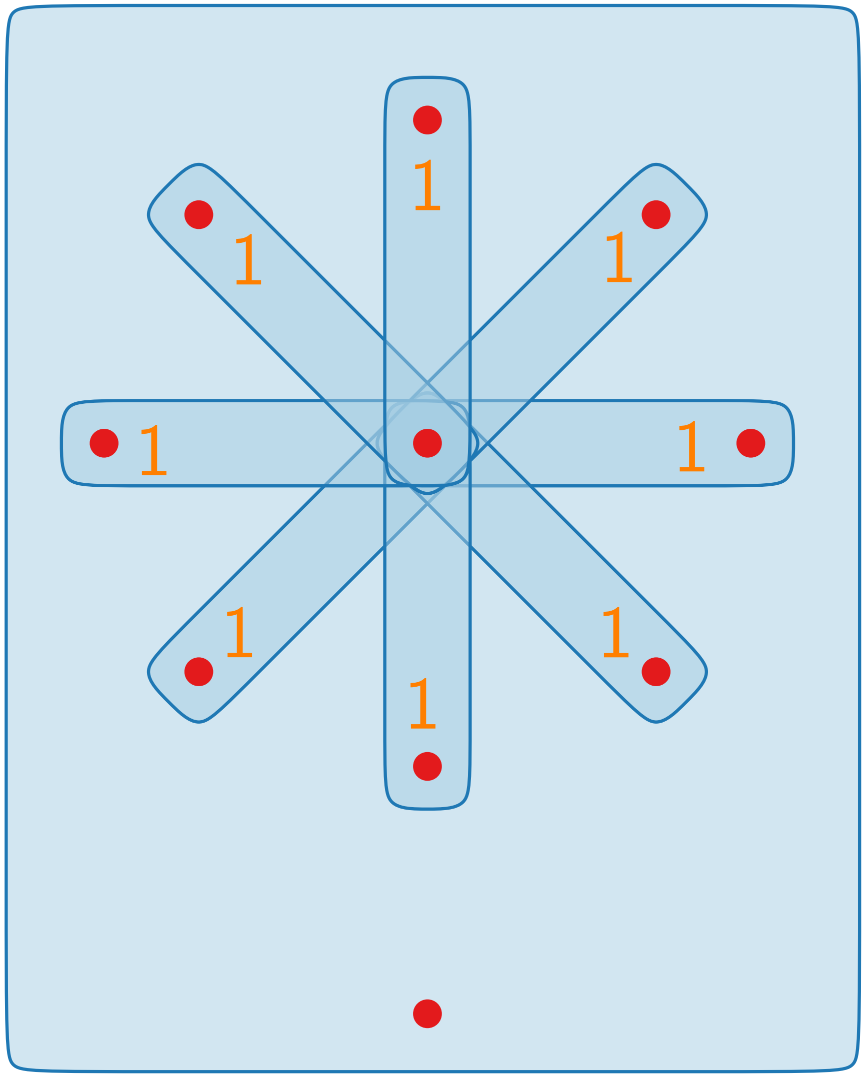
Tight Example



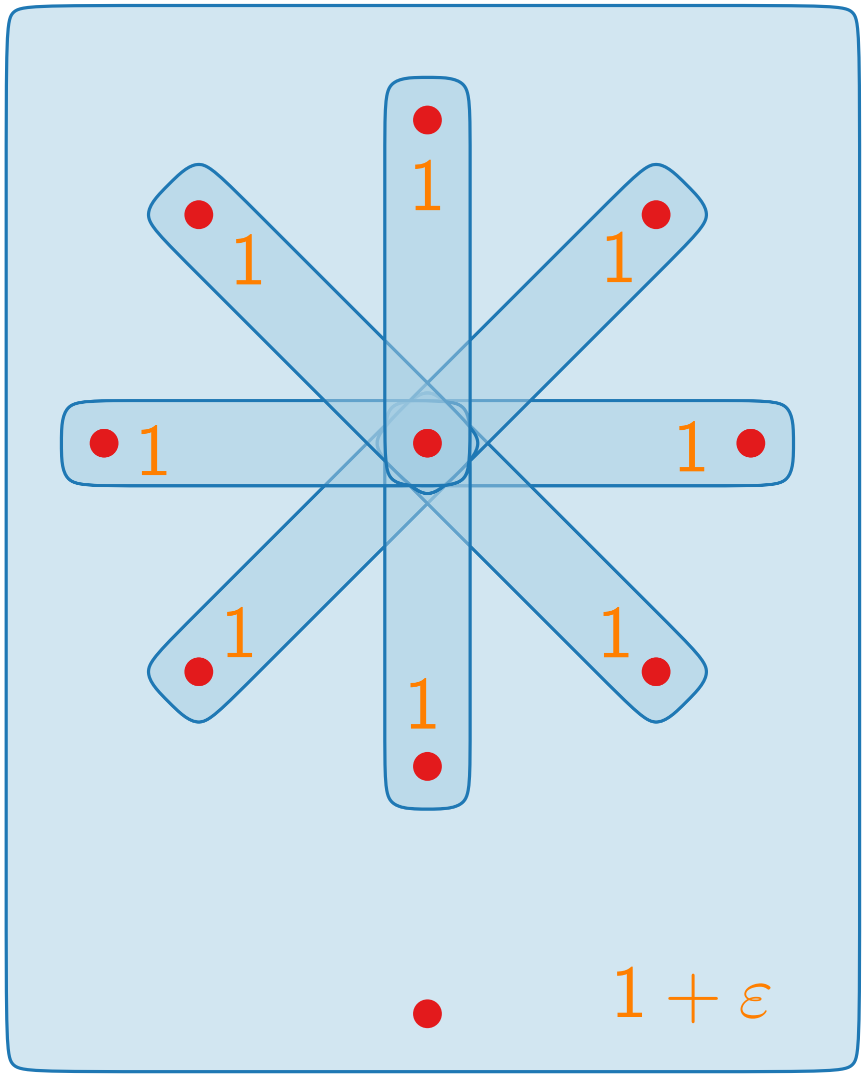
Tight Example



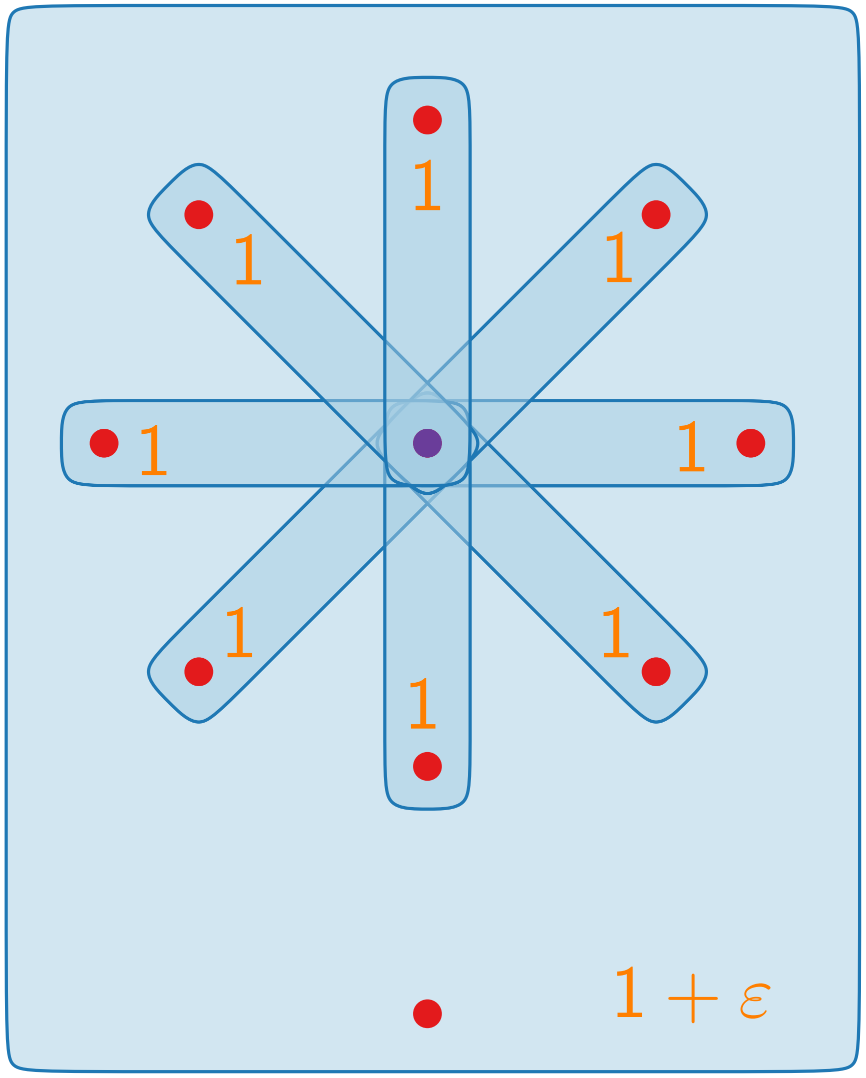
Tight Example



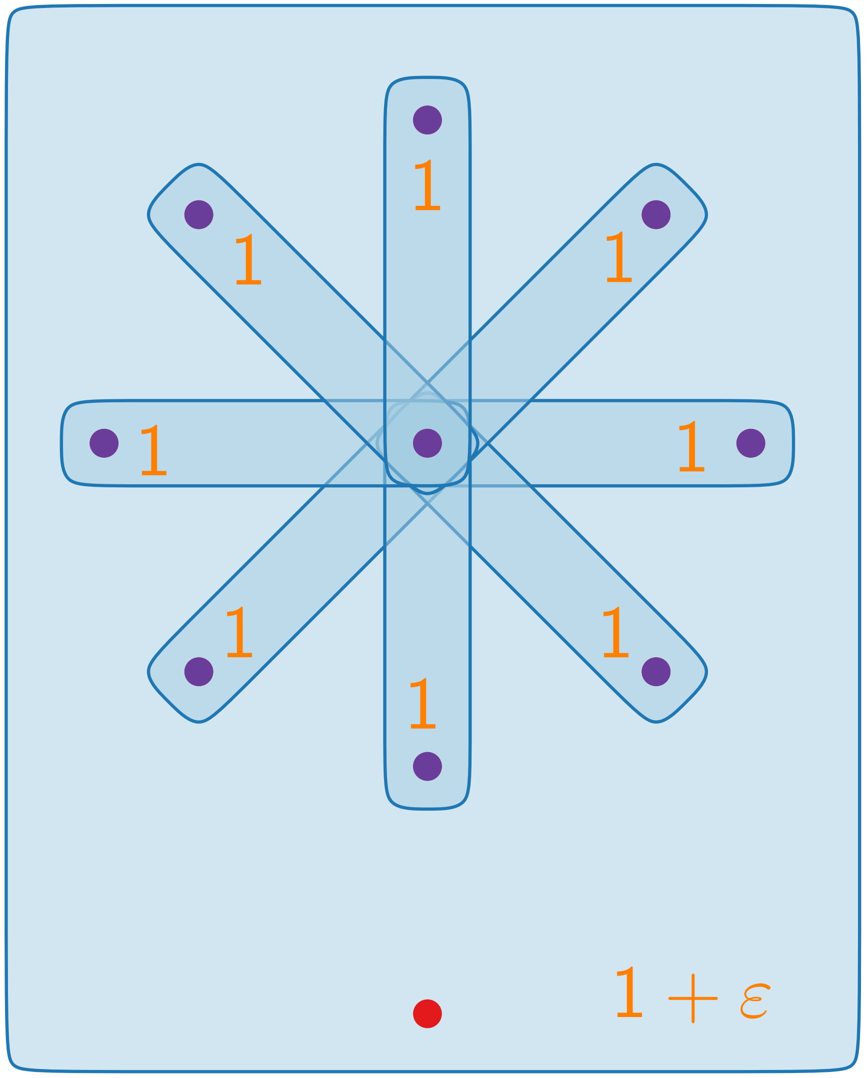
Tight Example



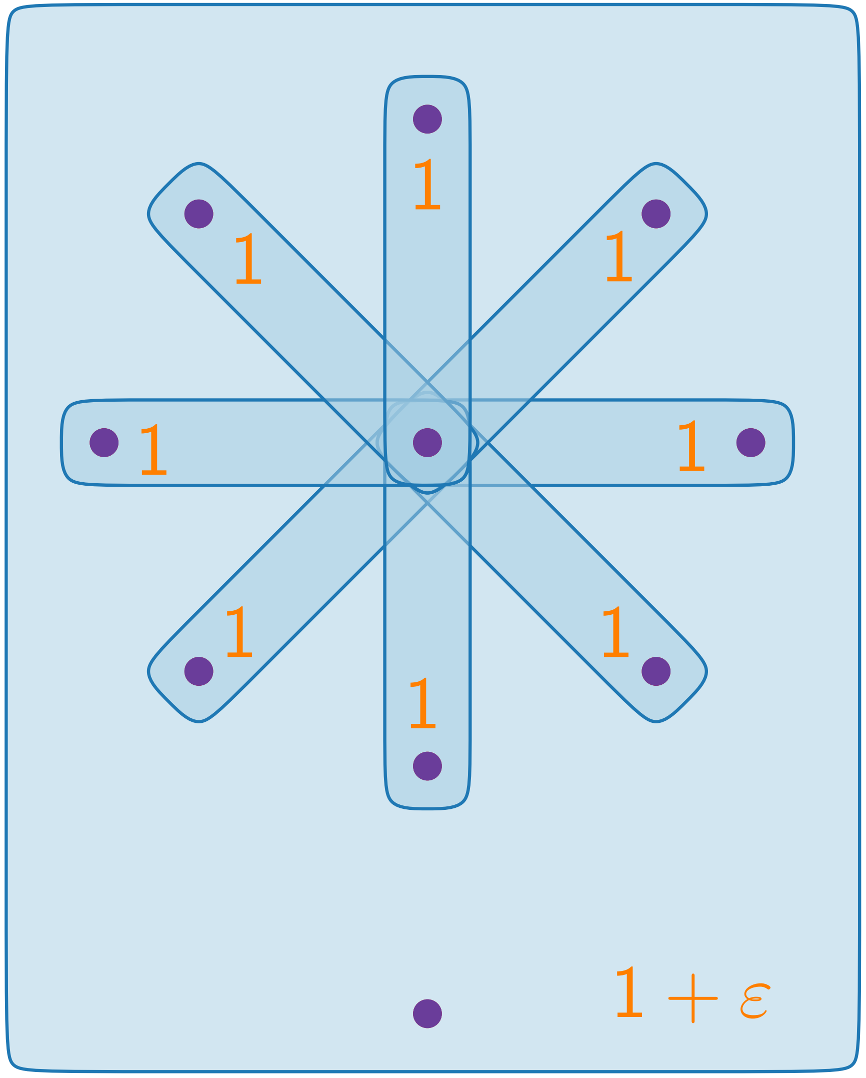
Tight Example



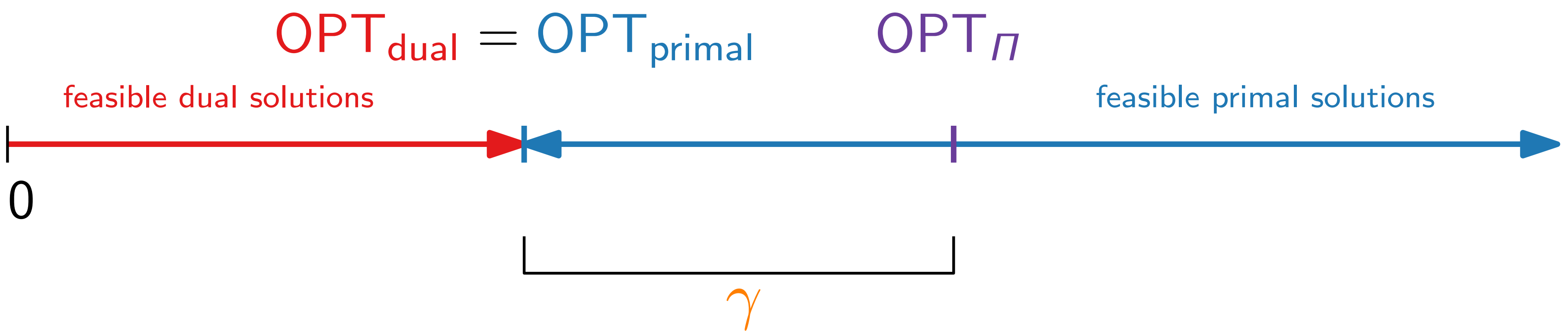
Tight Example



Tight Example

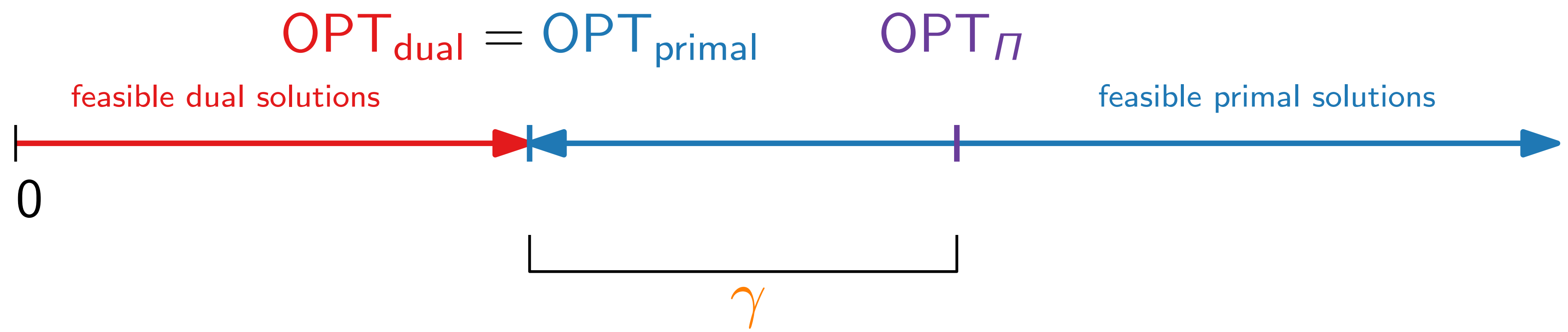


Integrality Gap



Consider a minimization problem Π in ILP form.

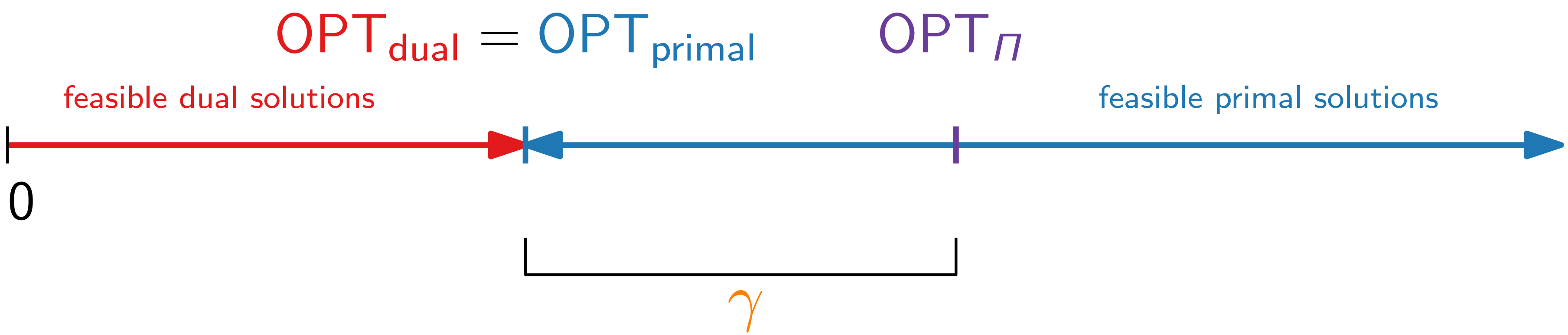
Integrality Gap



Consider a minimization problem Π in ILP form.

Dual methods (without outside help) are limited by the *integrality gap* of the LP-relaxation

Integrality Gap

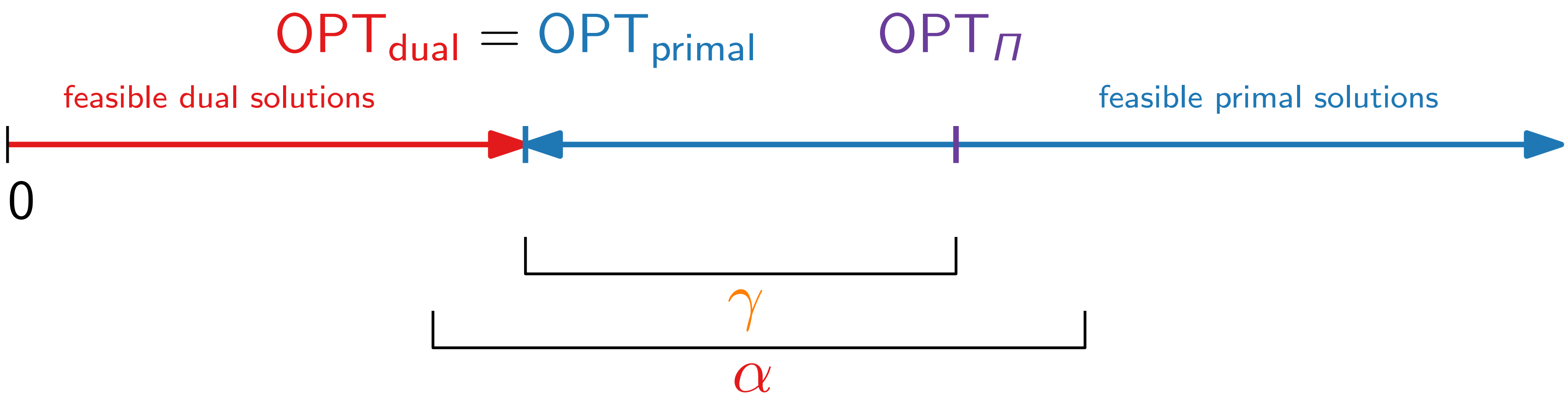


Consider a minimization problem Π in ILP form.

Dual methods (without outside help) are limited by the *integrality gap* of the LP-relaxation

$$\gamma = \sup_I \frac{\mathrm{OPT}_{\Pi}(I)}{\mathrm{OPT}_{\mathrm{primal}}(I)}$$

Integrality Gap



Consider a minimization problem Π in ILP form.

Dual methods (without outside help) are limited by the *integrality gap* of the LP-relaxation

$$\alpha \geq \gamma = \sup_{I} \frac{\mathrm{OPT}_{\Pi}(I)}{\mathrm{OPT}_{\mathrm{primal}}(I)}$$

Approximation Algorithms

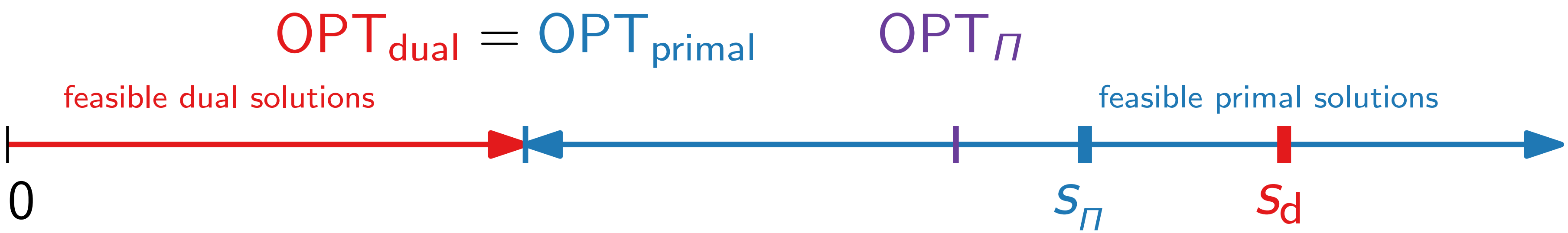
Lecture 5:
LP-based Approximation Algorithms for SetCover

Part IV:
Dual Fitting

Technique III) Dual Fitting

Consider a minimization problem Π in ILP form.

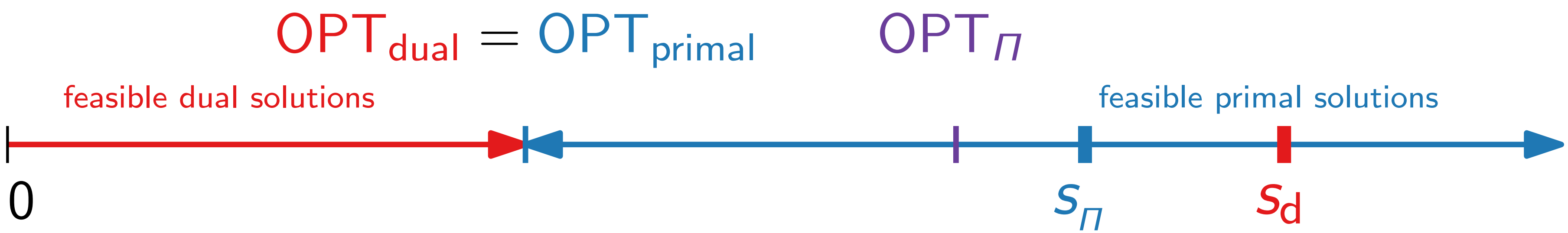
Technique III) Dual Fitting



Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π},

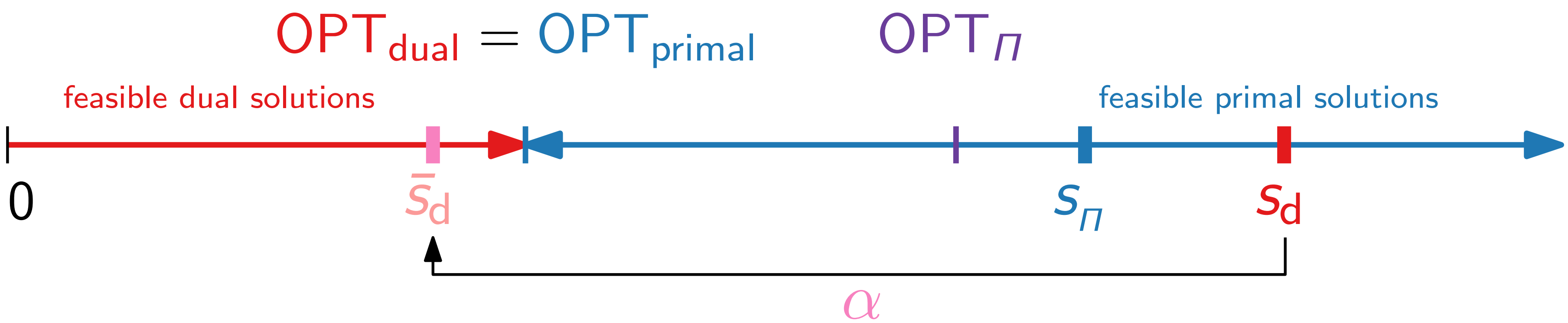
Technique III) Dual Fitting



Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π}, i.e., $\mathrm{obj}(s_{\Pi}) \leq \mathrm{obj}(s_{\mathrm{d}})$.

Technique III) Dual Fitting

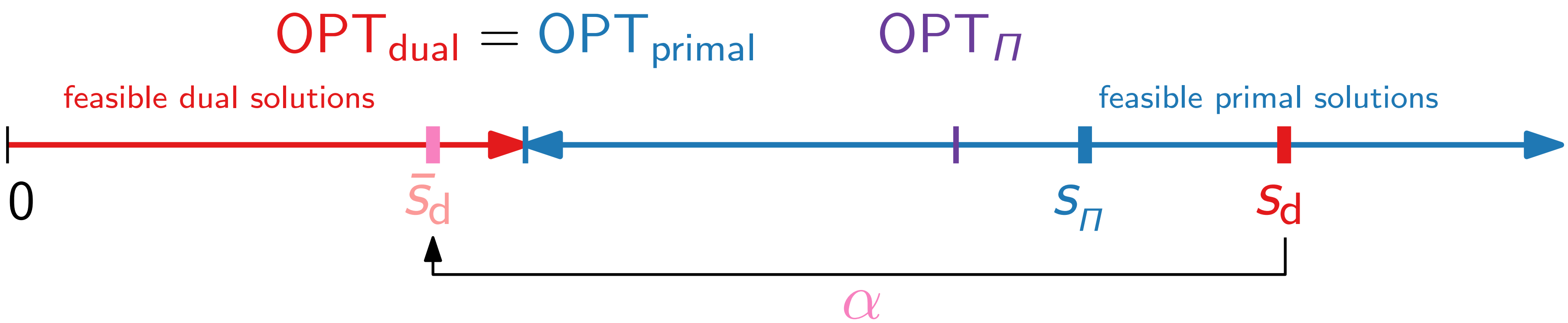


Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π}, i.e., $\mathrm{obj}(s_{\Pi}) \leq \mathrm{obj}(s_{\mathrm{d}})$.

Scale the dual variables $\rightsquigarrow$ feasible dual solution $\bar{s}_{\mathrm{d}}$.

Technique III) Dual Fitting



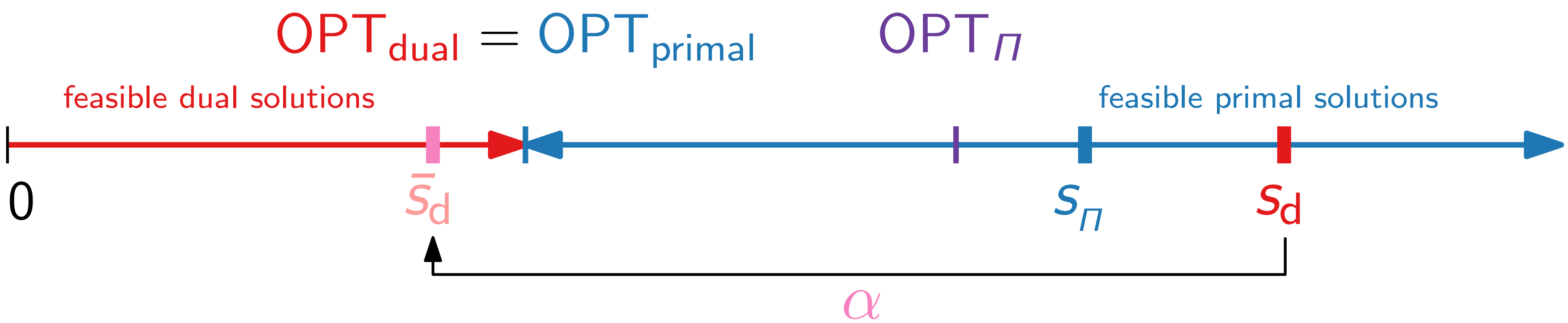
Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π}, i.e., $\mathrm{obj}(s_{\Pi}) \leq \mathrm{obj}(s_{\mathrm{d}})$.

Scale the dual variables $\rightsquigarrow$ feasible dual solution $\bar{s}_{\mathrm{d}}$.

$\Rightarrow$ $\mathrm{obj}(\bar{s}_{\mathrm{d}}) \leq \mathrm{OPT}_{\mathrm{dual}} \leq \mathrm{OPT}_{\Pi}$

Technique III) Dual Fitting



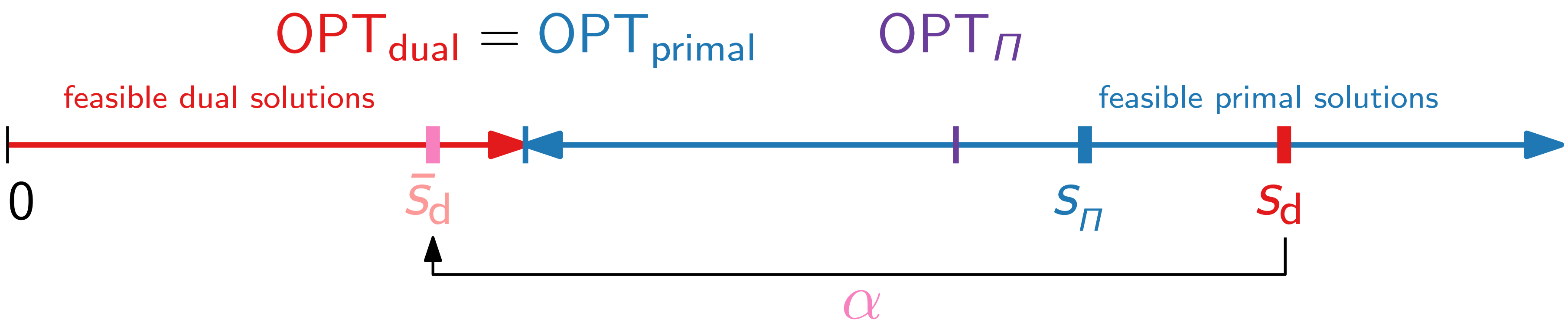
Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π}, i.e., $\mathrm{obj}(s_{\Pi}) \leq \mathrm{obj}(s_{\mathrm{d}})$.

Scale the dual variables $\rightsquigarrow$ feasible dual solution $\bar{s}_{\mathrm{d}}$.

$$\Rightarrow \qquad \mathrm{obj}(s_{\mathrm{d}})/\alpha = \mathrm{obj}(\bar{s}_{\mathrm{d}}) \leq \mathrm{OPT}_{\mathrm{dual}} \leq \mathrm{OPT}_{\Pi}$$

Technique III) Dual Fitting



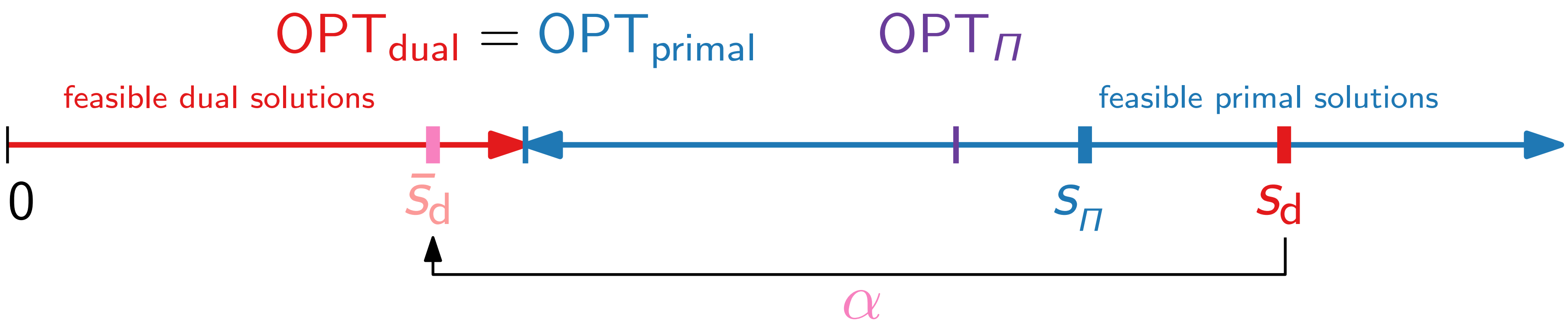
Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π}, i.e., $\mathrm{obj}(s_{\Pi}) \leq \mathrm{obj}(s_{\mathrm{d}})$.

Scale the dual variables $\rightsquigarrow$ feasible dual solution $\bar{s}_{\mathrm{d}}$.

$$\Rightarrow \mathrm{obj}(s_{\Pi})/\alpha \leq \mathrm{obj}(s_{\mathrm{d}})/\alpha = \mathrm{obj}(\bar{s}_{\mathrm{d}}) \leq \mathrm{OPT}_{\mathrm{dual}} \leq \mathrm{OPT}_{\Pi}$$

Technique III) Dual Fitting



Consider a minimization problem Π in ILP form.

Combinatorial algorithm (e.g., greedy) computes feasible primal solution s_{Π} and infeasible dual solution s_{d} that completely "pays" for s_{Π}, i.e., $\mathrm{obj}(s_{\Pi}) \le \mathrm{obj}(s_{\mathrm{d}})$.

Scale the dual variables $\rightsquigarrow$ feasible dual solution $\bar{s}_{\mathrm{d}}$.

$\Rightarrow \mathrm{obj}(s_{\Pi})/\alpha \le \mathrm{obj}(s_{\mathrm{d}})/\alpha = \mathrm{obj}(\bar{s}_{\mathrm{d}}) \le \mathrm{OPT}_{\mathrm{dual}} \le \mathrm{OPT}_{\Pi}$

$\Rightarrow$ Scaling factor α is approximation factor :-)

Dual Fitting for SetCover

Combinatorial (greedy) algorithm (see Lecture #2):

```
GreedySetCover(U, 𝒮, c)
  C ← ∅
  𝒮' ← ∅
  while C ≠ U do
      S ← Set from 𝒮 that minimizes c(S)/|S \ C|
      foreach u ∈ S \ C do
          price(u) ← c(S)/|S \ C|
      C ← C ∪ S
      𝒮' ← 𝒮' ∪ {S}
  return 𝒮'                                  // Cover of U
```

Dual Fitting for SetCover

Combinatorial (greedy) algorithm (see Lecture #2):

GreedySetCover($U, \mathcal{S}, c$)

$C \leftarrow \emptyset$

$\mathcal{S}' \leftarrow \emptyset$

while $C \neq U$ **do**

  $S \leftarrow$ Set from $\mathcal{S}$ that minimizes $\frac{c(S)}{|S \setminus C|}$

  foreach $u \in S \setminus C$ **do**

    $\text{price}(u) \leftarrow \frac{c(S)}{|S \setminus C|}$

  $C \leftarrow C \cup S$

  $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{S\}$

return $\mathcal{S}'$ // Cover of U

Reminder: $\sum_{u \in U} \text{price}(u)$ ...

Dual Fitting for SetCover

Combinatorial (greedy) algorithm (see Lecture #2):

```
GreedySetCover(U, 𝒮, c)
  C ← ∅
  𝒮' ← ∅
  while C ≠ U do
      S ← Set from 𝒮 that minimizes c(S)/|S \ C|
      foreach u ∈ S \ C do
          price(u) ← c(S)/|S \ C|
      C ← C ∪ S
      𝒮' ← 𝒮' ∪ {S}
  return 𝒮'                                  // Cover of U
```

Reminder: $\sum_{u \in U} \text{price}(u)$ completely pays for $\mathcal{S}'$.

New: LP-based Analysis

Observation. For each $u \in U$, $\mathrm{price}(u)$ is a dual variable

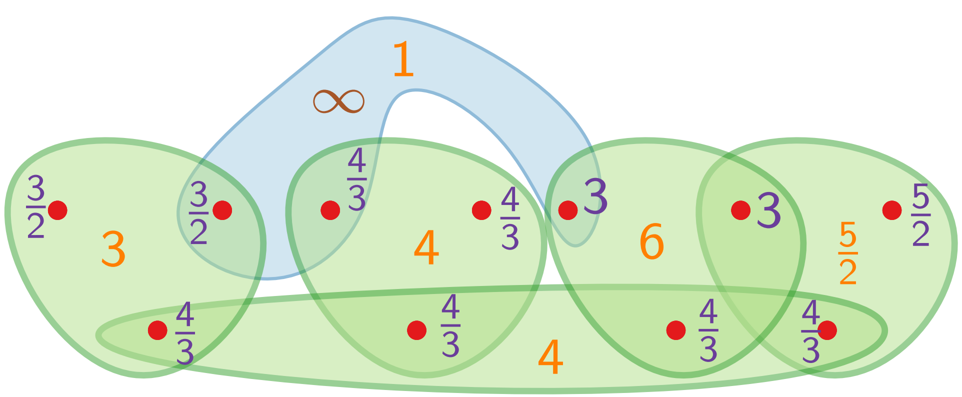
New: LP-based Analysis

Observation. For each $u \in U$, $\mathrm{price}(u)$ is a dual variable

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

New: LP-based Analysis

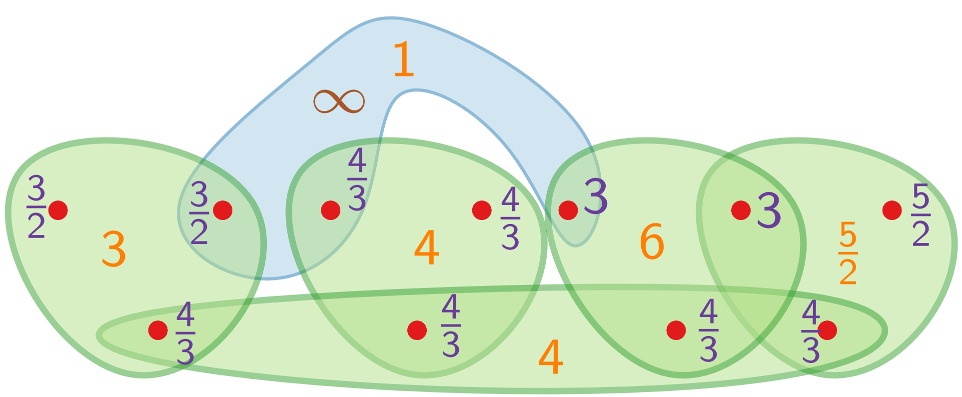
Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable



$$\begin{aligned} &\textbf{maximize} && \sum_{u \in U} y_u \\ &\textbf{subject to} && \sum_{u \in S} y_u \le c_S && S \in \mathcal{S} \\ & && y_u \ge 0 && u \in U \end{aligned}$$

New: LP-based Analysis

Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

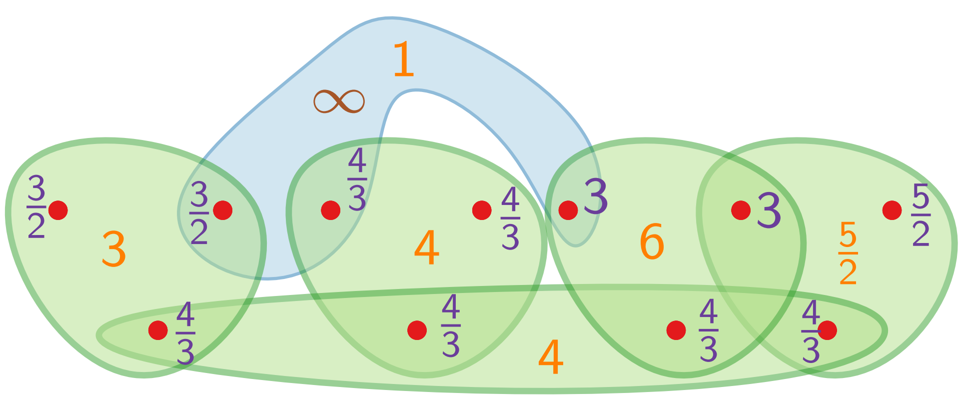


$$
\begin{array}{lll}
\textbf{maximize} & \sum_{u \in U} y_u & \\
\textbf{subject to} & \sum_{u \in S} y_u \le c_S & S \in \mathcal{S} \\
 & y_u \ge 0 & u \in U
\end{array}
$$

New: LP-based Analysis

Observation. For each $u \in U$, $\mathrm{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.



$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \le c_S & S \in \mathcal{S} \\ & y_u \ge 0 & u \in U \end{array}$$

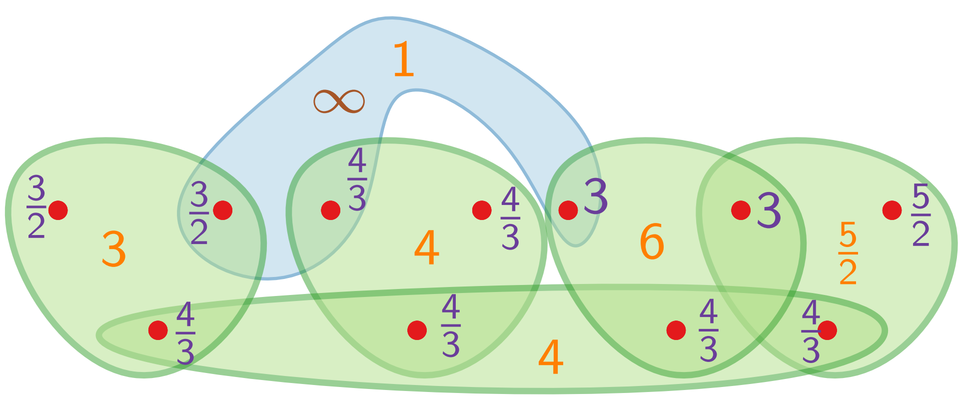
New: LP-based Analysis

Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.



maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

New: LP-based Analysis

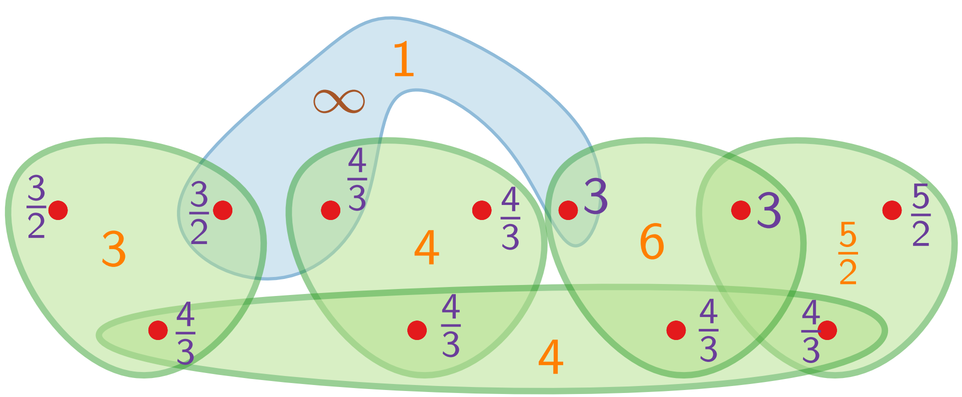
Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.

Take $\bar{y}_u =$



maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

New: LP-based Analysis

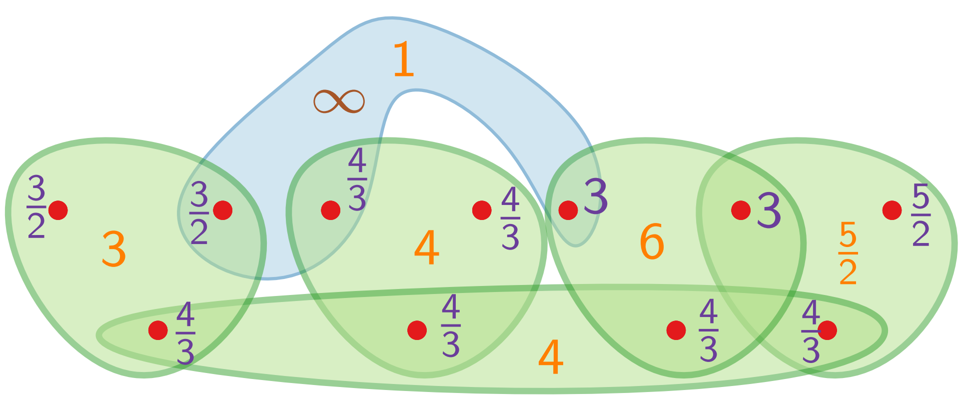
Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.

Take $\bar{y}_u = \ \text{price}(u)/$



$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\
\textbf{subject to} & \displaystyle\sum_{u \in S} y_u \le c_S & S \in \mathcal{S} \\
& y_u \ge 0 & u \in U
\end{array}
$$

New: LP-based Analysis

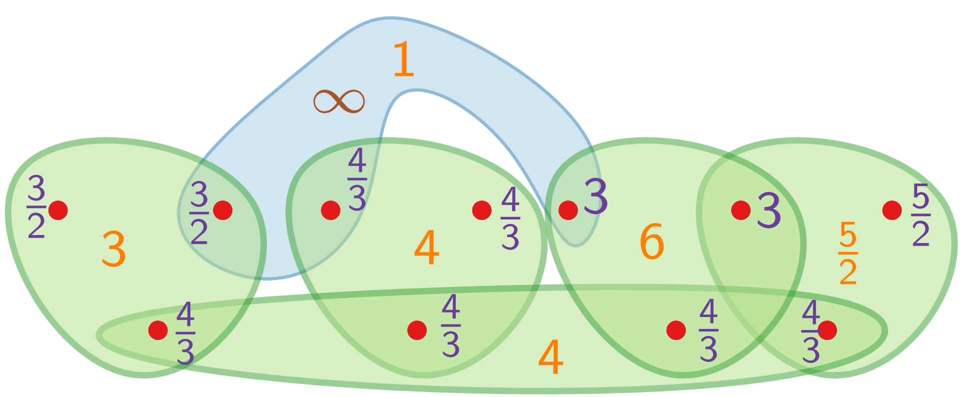
Observation. For each $u \in U$, price(u) is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.

Take $\bar{y}_u = \text{price}(u) / \mathcal{H}_k$.



$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\
\textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\
& y_u \geq 0 & u \in U
\end{array}
$$

New: LP-based Analysis

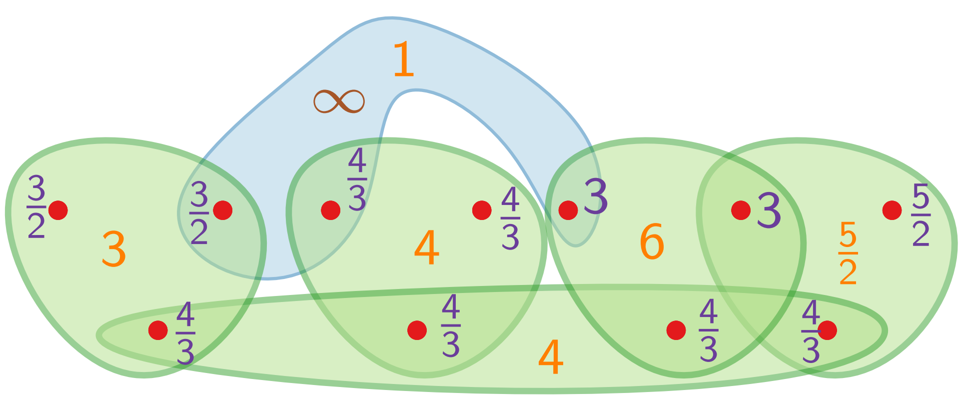
Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are “overpacked” by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.

Take $\bar{y}_u = \text{price}(u) / \mathcal{H}_k$. ($k =$ cardinality of largest set in $\mathcal{S}$.)



maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \le c_S$	$S \in \mathcal{S}$
	$y_u \ge 0$	$u \in U$

New: LP-based Analysis

Observation. For each $u \in U$, $\mathrm{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

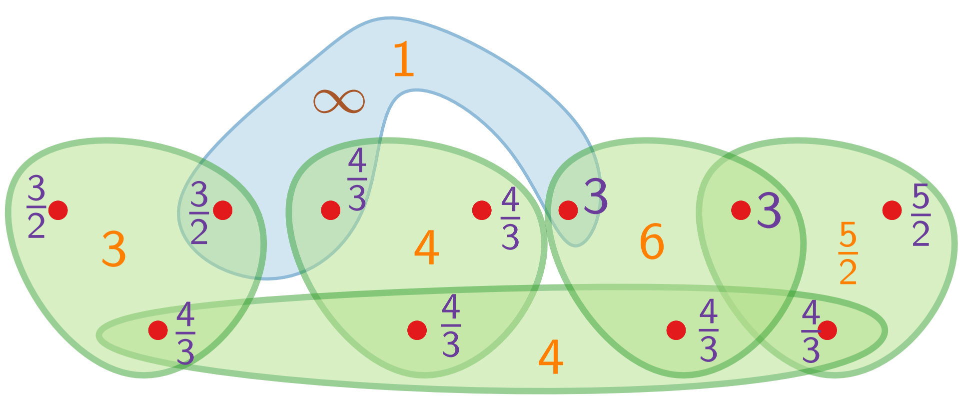
Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.

Take $\bar{y}_u = \mathrm{price}(u) / \mathcal{H}_k$. ($k =$ cardinality of largest set in $\mathcal{S}$.)

The greedy algorithm uses *these* dual variables as lower bound for OPT.



maximize $\sum_{u \in U} y_u$

subject to $\sum_{u \in S} y_u \leq c_S \quad S \in \mathcal{S}$

$y_u \geq 0 \quad u \in U$

New: LP-based Analysis

Observation. For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

Homework exercise: Construct instance where some S are "overpacked" by factor $\approx H_{|S|}$.

Dual-fitting trick:

Scale dual variables such that no set is overpacked.

Take $\bar{y}_u = \ \text{price}(u)/\ \mathcal{H}_k$. $(k =$ cardinality of largest set in $\mathcal{S}$.)

The greedy algorithm uses *these* dual variables as lower bound for OPT.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\
\textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\
 & y_u \geq 0 & u \in U
\end{array}
$$

Proof.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

Proof. To prove: No set is overpacked by $\bar{y}$.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$
\begin{array}{lll}
\textbf{maximize} & \sum_{u \in U} y_u & \\
\textbf{subject to} & \sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\
 & y_u \geq 0 & u \in U
\end{array}
$$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$\begin{array}{lll} \textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\ \textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\ & y_u \geq 0 & u \in U \end{array}$$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \ldots, u_\ell$ be the elements of S –

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\
\textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\
 & y_u \geq 0 & u \in U
\end{array}
$$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \ldots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$
\begin{array}{lll}
\textbf{maximize} & \displaystyle\sum_{u \in U} y_u & \\
\textbf{subject to} & \displaystyle\sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\
& y_u \geq 0 & u \in U
\end{array}
$$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \ldots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \ldots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \leq$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \leq \; c(S)/(\ell - i + 1)$.

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

$$\begin{array}{lll}
\textbf{maximize} & \sum_{u \in U} y_u & \\
\textbf{subject to} & \sum_{u \in S} y_u \leq c_S & S \in \mathcal{S} \\
& y_u \geq 0 & u \in U
\end{array}$$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \ldots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \leq c(S)/(\ell - i + 1)$.

$\Rightarrow \bar{y}_{u_i} \leq$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\mathrm{price}(u_i) \leq c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1}$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\mathrm{price}(u_i) \leq \; c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \leq$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \le k$.

Let $u_1, \dots, u_\ell$ be the elements of S – in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\ge \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \le \; c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \le \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \;\Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \le \frac{c(S)}{\mathcal{H}_k} \cdot (\qquad\qquad)$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \le c_S$	$S \in \mathcal{S}$
	$y_u \ge 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\mathrm{price}(u_i) \leq \; c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \;\Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \left(\frac{1}{\ell} + \cdots + \frac{1}{1}\right)$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \le k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\ge \ell - i + 1$ elem. of S are uncovered.

So $\mathrm{price}(u_i) \le \; c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \le \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \;\Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \le \frac{c(S)}{\mathcal{H}_k} \cdot \overbrace{\left(\frac{1}{\ell} + \dots + \frac{1}{1}\right)}$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \le c_S$	$S \in \mathcal{S}$
	$y_u \ge 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \leq \ c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \ \Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \overbrace{\left(\tfrac{1}{\ell} + \dots + \tfrac{1}{1}\right)}^{= \mathcal{H}_\ell}$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \leq \; c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \;\Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \overbrace{\left(\tfrac{1}{\ell} + \cdots + \tfrac{1}{1}\right)}^{= \mathcal{H}_\ell \leq \mathcal{H}_k}$$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Proof. To prove: No set is overpacked by $\bar{y}$.

Let $S \in \mathcal{S}$ and $\ell = |S| \leq k$.

Let $u_1, \dots, u_\ell$ be the elements of S –
in the order in which they are covered by greedy.

Consider the iteration in which u_i is covered.

Before that, $\geq \ell - i + 1$ elem. of S are uncovered.

So $\text{price}(u_i) \leq c(S)/(\ell - i + 1)$.

$$\Rightarrow \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \frac{1}{\ell - i + 1} \Rightarrow \sum_{i=1}^{\ell} \bar{y}_{u_i} \leq \frac{c(S)}{\mathcal{H}_k} \cdot \overbrace{\left(\frac{1}{\ell} + \dots + \frac{1}{1}\right)}^{= \mathcal{H}_\ell \leq \mathcal{H}_k} \leq c(S)$$

$\square$

Lemma.
The vector $\bar{y} = (\bar{y}_u)_{u \in U}$ is a feasible solution for the dual LP.

maximize	$\sum_{u \in U} y_u$	
subject to	$\sum_{u \in S} y_u \leq c_S$	$S \in \mathcal{S}$
	$y_u \geq 0$	$u \in U$

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof. $\mathsf{ALG} = c(\mathcal{S}') \leq$

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof. $\mathsf{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \mathrm{price}(u) =$

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof. $\mathrm{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \mathrm{price}(u) = \mathcal{H}_k \cdot \sum_{u \in U} \bar{y}_u \leq$

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof. $$\mathrm{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \mathrm{price}(u) = \mathcal{H}_k \cdot \sum_{u \in U} \bar{y}_u \leq$$

$$\leq \mathcal{H}_k \cdot \mathrm{OPT}_{\mathrm{relax}}$$

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof.

$$\begin{aligned}
\mathrm{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \mathrm{price}(u) = \mathcal{H}_k \cdot \sum_{u \in U} \bar{y}_u &\leq \\
&\leq \mathcal{H}_k \cdot \mathrm{OPT}_{\mathrm{relax}} \\
&\leq \mathcal{H}_k \cdot \mathrm{OPT} \qquad \square
\end{aligned}$$

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof.

$$\begin{aligned}\mathsf{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \mathrm{price}(u) = \mathcal{H}_k \cdot \sum_{u \in U} \bar{y}_u &\leq \\ &\leq \mathcal{H}_k \cdot \mathsf{OPT}_{\mathsf{relax}} \\ &\leq \mathcal{H}_k \cdot \mathsf{OPT} \qquad \square\end{aligned}$$

Strengthened bound with respect to $\mathsf{OPT}_{\mathsf{relax}} \leq \mathsf{OPT}$.

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof.

$$\begin{aligned} \mathrm{ALG} = c(\mathcal{S}') \le \sum_{u \in U} \mathrm{price}(u) = \mathcal{H}_k \cdot \sum_{u \in U} \bar{y}_u &\le \\ \le \mathcal{H}_k \cdot \mathrm{OPT}_{\mathrm{relax}} & \\ \le \mathcal{H}_k \cdot \mathrm{OPT} & \qquad \square \end{aligned}$$

Strengthened bound with respect to $\mathrm{OPT}_{\mathrm{relax}} \le \mathrm{OPT}$.

Dual solution allows a *per-instance* estimation $c(\mathcal{S}')/\mathrm{OPT}_{\mathrm{relax}}$ of the quality of the greedy solution

Result for Dual Fitting

Theorem. GreedySetCover is a factor-$\mathcal{H}_k$ approximation algorithm for $\textsc{SetCover}$, where $k = \max_{S \in \mathcal{S}} |S|$.

Proof.

$$\begin{aligned} \mathrm{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \mathrm{price}(u) = \mathcal{H}_k \cdot \sum_{u \in U} \bar{y}_u &\leq \\ &\leq \mathcal{H}_k \cdot \mathrm{OPT}_{\mathrm{relax}} \\ &\leq \mathcal{H}_k \cdot \mathrm{OPT} \qquad \square \end{aligned}$$

Strengthened bound with respect to $\mathrm{OPT}_{\mathrm{relax}} \leq \mathrm{OPT}$.

Dual solution allows a *per-instance* estimation $c(\mathcal{S}')/\mathrm{OPT}_{\mathrm{relax}}$ of the quality of the greedy solution

...which may be stronger than the worst-case bound $\mathcal{H}_k$.