# Approximation Algorithms

## Lecture 5:
## LP-based Approximation Algorithms for SetCover

### Part I:
### LP-based Approximation Techniques

Philipp Kindermann

Summer Semester 2020

# I) LP-Rounding


Consider a minimization problem $\Pi$ in ILP-form.

# I) LP-Rounding


Consider a minimization problem $\Pi$ in ILP-form.

Compute a solution for the LP-relaxation

# I) LP-Rounding


Consider a minimization problem $\Pi$ in ILP-form.

Compute a solution for the LP-relaxation

Round to obtain an integer solution for $\Pi$

# I) LP-Rounding


Consider a minimization problem $\Pi$ in ILP-form.

Compute a solution for the LP-relaxation

Round to obtain an integer solution for $\Pi$

Difficulty: ensure **feasible** solution of $\Pi$

# I) LP-Rounding


Consider a minimization problem $\Pi$ in ILP-form.

Compute a solution for the LP-relaxation

Round to obtain an integer solution for $\Pi$

Difficulty: ensure **feasible** solution of $\Pi$

Approximation factor $ALG/OPT_{\Pi} \leq ALG/OPT_{relax}$

# II) Primal-Dual Approach


Consider a minimization problem $\Pi$ in ILP-form.

# II) Primal-Dual Approach


Consider a minimization problem $\Pi$ in ILP-form.

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

# II) Primal-Dual Approach


Consider a minimization problem $\Pi$ in ILP-form.

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

# II) Primal-Dual Approach


Consider a minimization problem $\Pi$ 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 $s_{\Pi}$ for $\Pi$ iteratively:
increase $s_d$ according to CS and make $s_{\Pi}$ "more feasible".

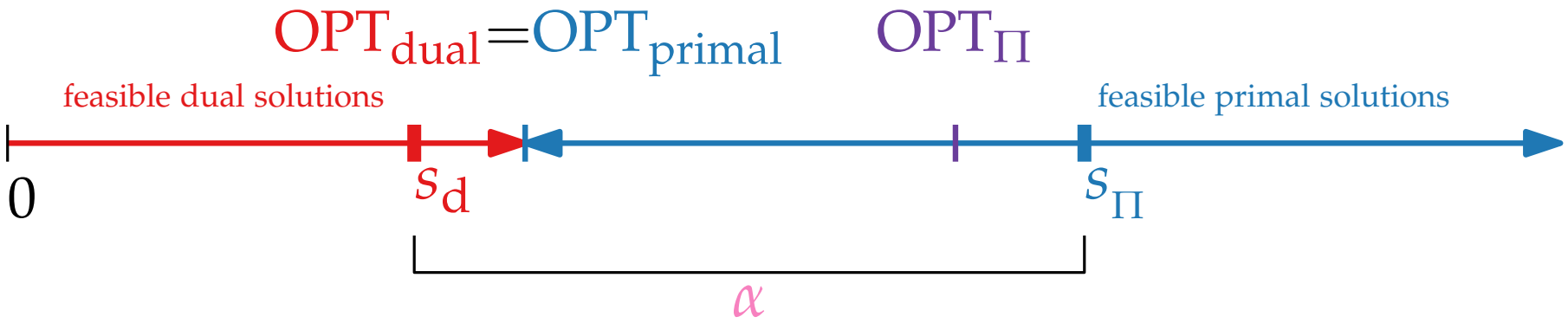
# II) Primal-Dual Approach


Consider a minimization problem $\Pi$ 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 $s_{\Pi}$ for $\Pi$ iteratively:
increase $s_d$ according to CS and make $s_{\Pi}$ "more feasible".

# II) Primal-Dual Approach



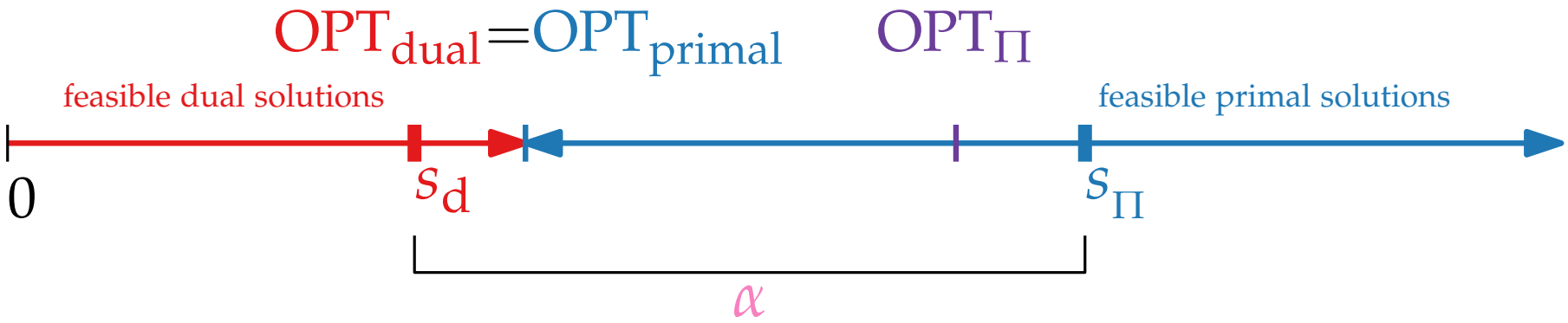

Consider a minimization problem $\Pi$ in ILP-form.

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

Compute dual solution $s_d$ and integral primal solution $s_{\Pi}$ for $\Pi$ iteratively:
increase $s_d$ according to CS and make $s_{\Pi}$ "more feasible".

Approximation factor $\leq \text{obj}(s_{\Pi}) / \text{obj}(s_d)$

# II) Primal-Dual Approach




Consider a minimization problem $\Pi$ 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 $s_{\Pi}$ for $\Pi$ iteratively:
increase $s_d$ according to CS and make $s_{\Pi}$ "more feasible".

Approximation factor $\leq \text{obj}(s_{\Pi})/\text{obj}(s_d)$

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

# III) Dual Fitting


Consider a minimization problem $\Pi$ in ILP-form.

# III) Dual Fitting


Consider a minimization problem $\Pi$ in ILP-form.

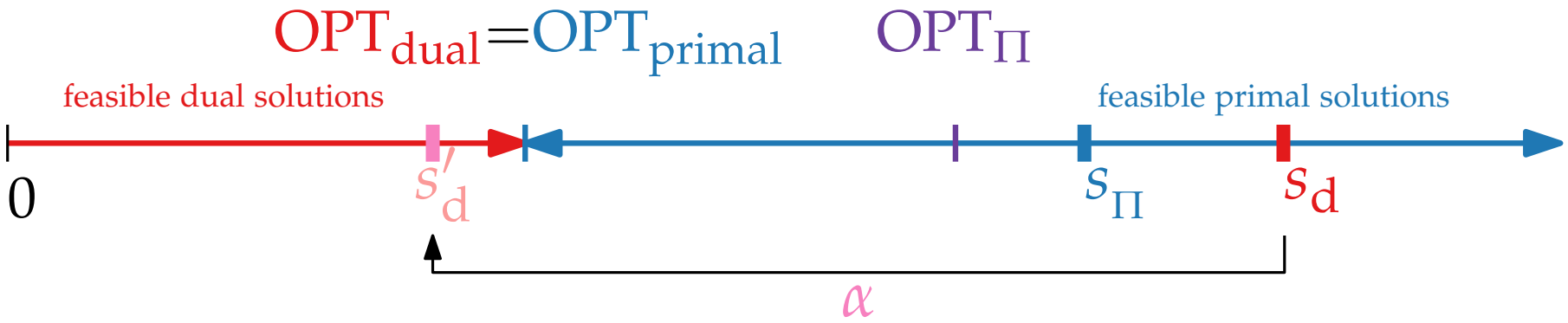
Combinatorial algorithm (e.g., greedy) computes feasible primal solution $s_{\Pi}$ and infeasible dual solution $s_d$ which "completely pays for" $s_{\Pi}$,

# III) Dual Fitting


Consider a minimization problem $\Pi$ in ILP-form.

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

# III) Dual Fitting

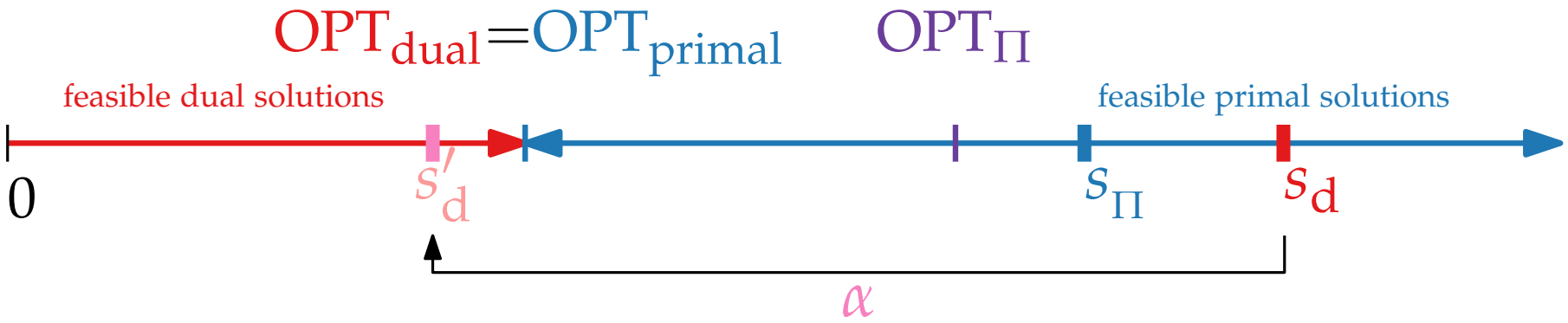



Consider a minimization problem $\Pi$ in ILP-form.

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

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

# III) Dual Fitting



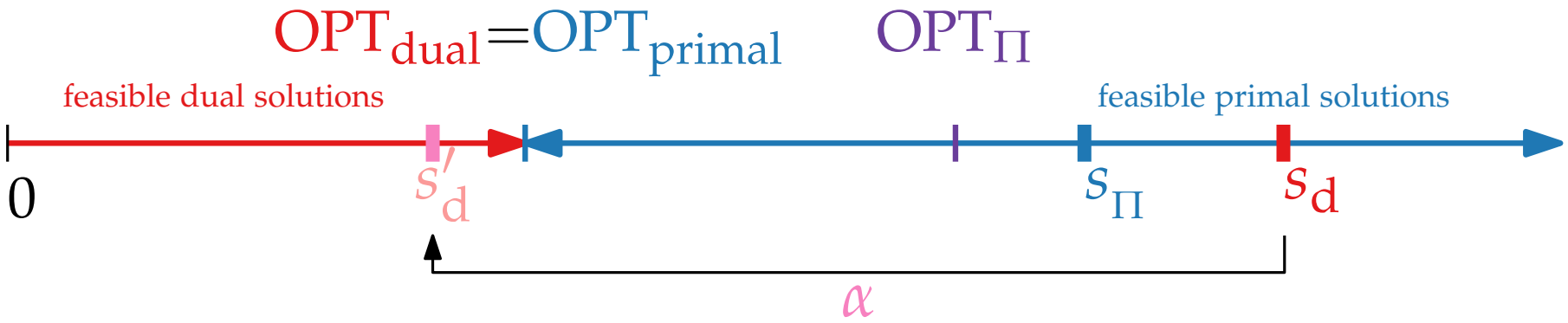

Consider a minimization problem $\Pi$ in ILP-form.

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

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

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

# III) Dual Fitting



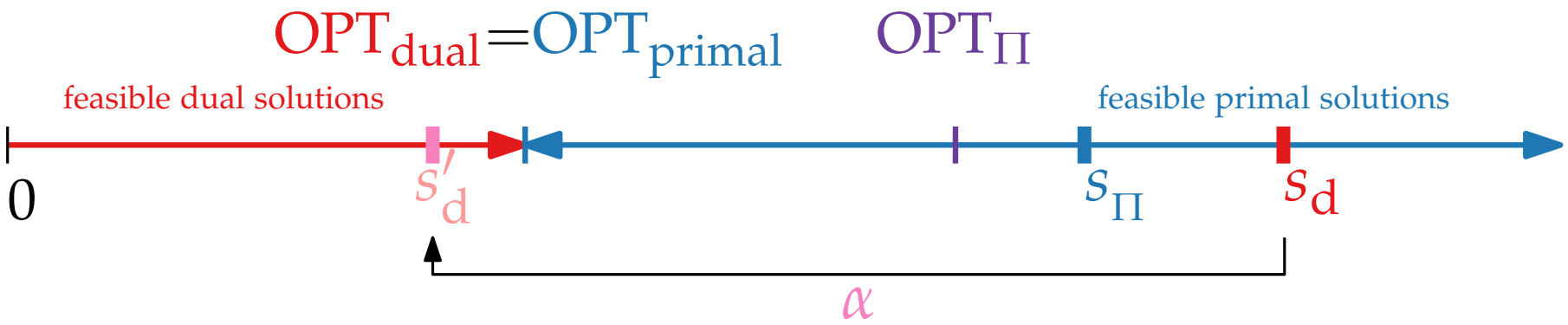

Consider a minimization problem $\Pi$ in ILP-form.

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

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

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

# III) Dual Fitting



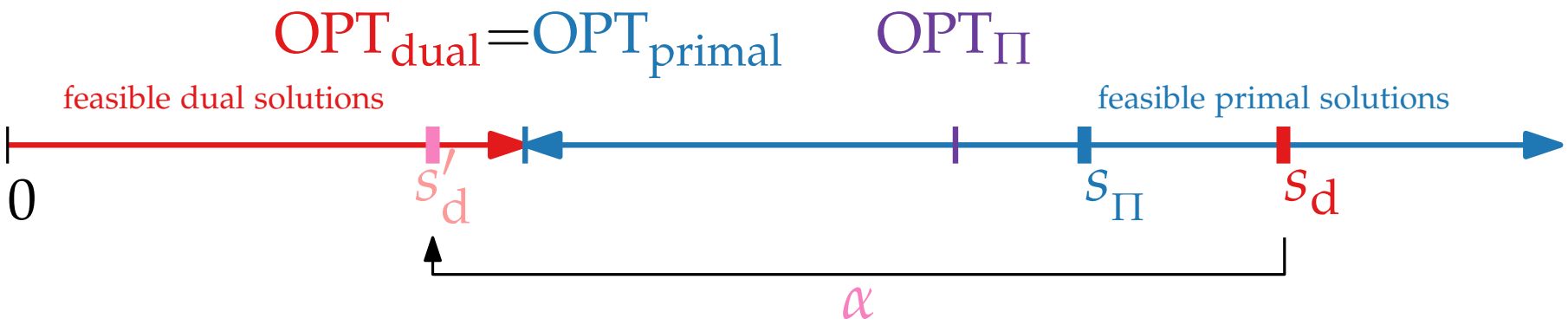

Consider a minimization problem $\Pi$ in ILP-form.

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

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

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

# III) Dual Fitting




Consider a minimization problem $\Pi$ in ILP-form.

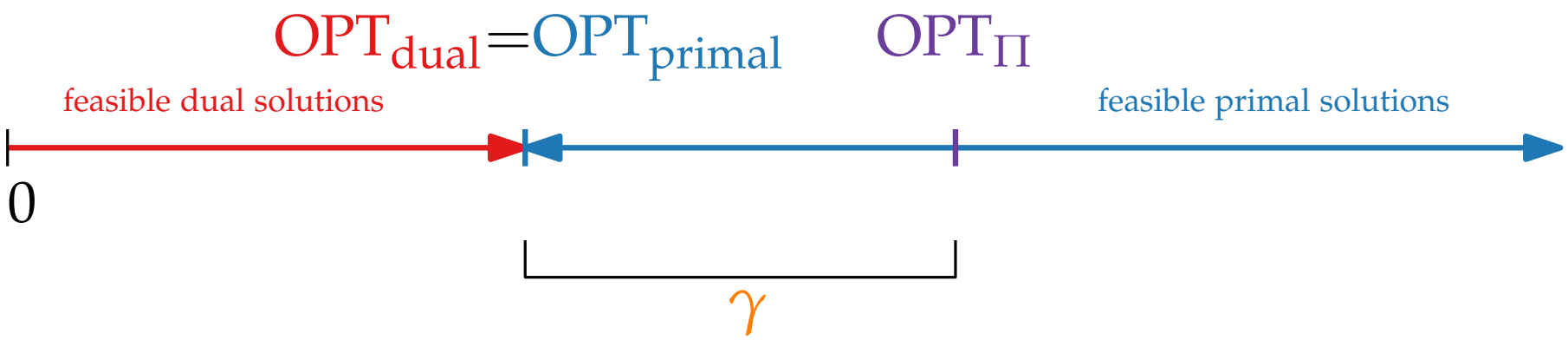
Combinatorial algorithm (e.g., greedy) computes feasible primal solution $s_{\Pi}$ and infeasible dual solution $s_d$ which "completely pays for" $s_{\Pi}$, i.e. $\text{obj}(s_{\Pi}) \leq \text{obj}(s_d)$.

Scale the dual variables $\rightsquigarrow$ feasible dual solution $s'_d$.

$\Rightarrow \text{obj}(s_{\Pi})/\alpha \leq \text{obj}(s_d)/\alpha = \text{obj}(s'_d) \leq OPT_{dual} \leq OPT_{\Pi}$

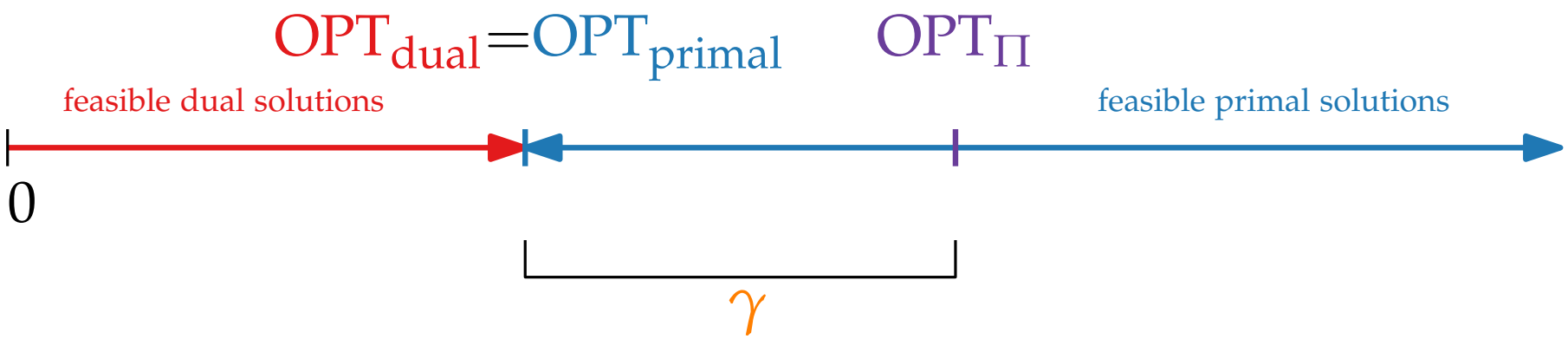
$\Rightarrow$ Scaling factor $\alpha$ is approximation factor.

# Integrality Gap




Consider a minimization problem $\Pi$ in ILP-form.

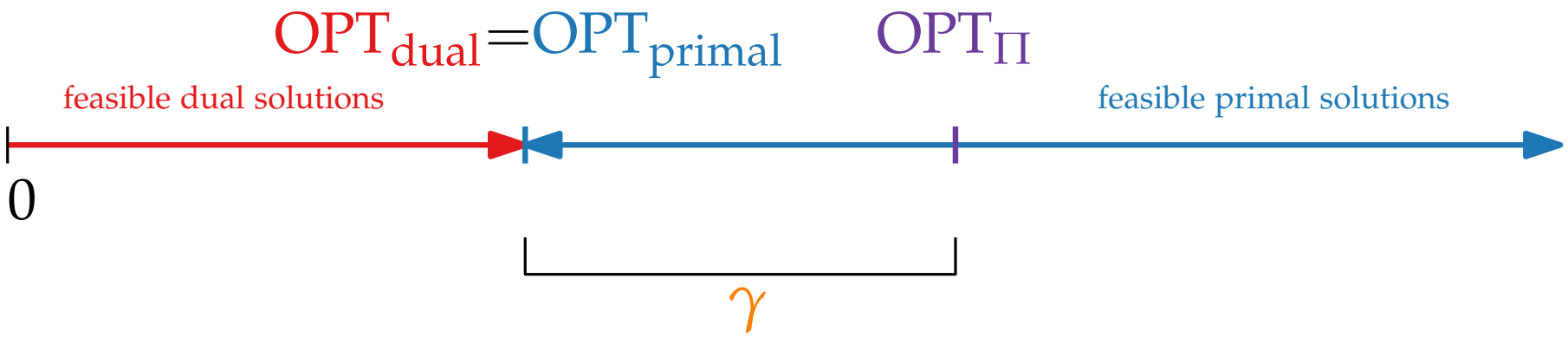
# Integrality Gap




Consider a minimization problem $\Pi$ in ILP-form.

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

# Integrality Gap

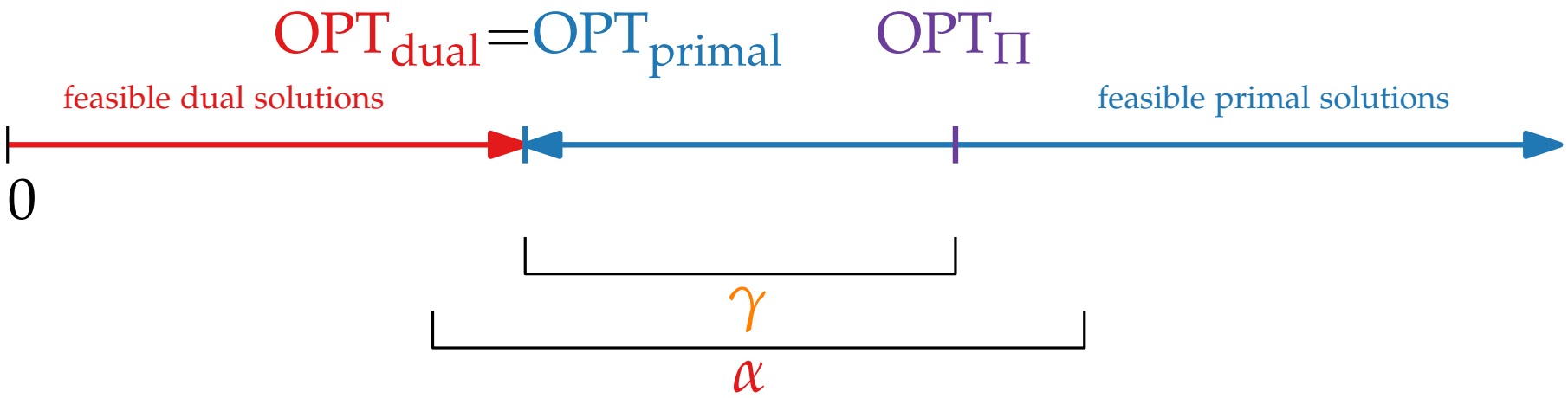



Consider a minimization problem $\Pi$ in ILP-form.

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

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

# Integrality Gap




Consider a minimization problem $\Pi$ in ILP-form.

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

$$\alpha \geq \gamma = \sup_{I} \frac{OPT_{\Pi}(I)}{OPT_{primal}(}$$

# Approximation Algorithms

## Lecture 5:
## LP-based Approximation Algorithms for SetCover

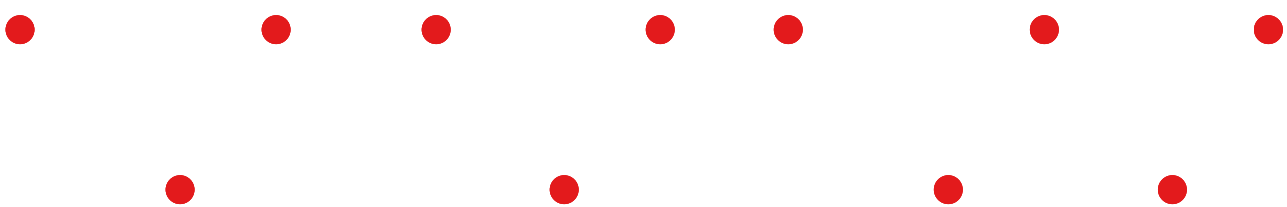
### Part II:
### SetCover as an ILP

Philipp Kindermann

Summer Semester 2020

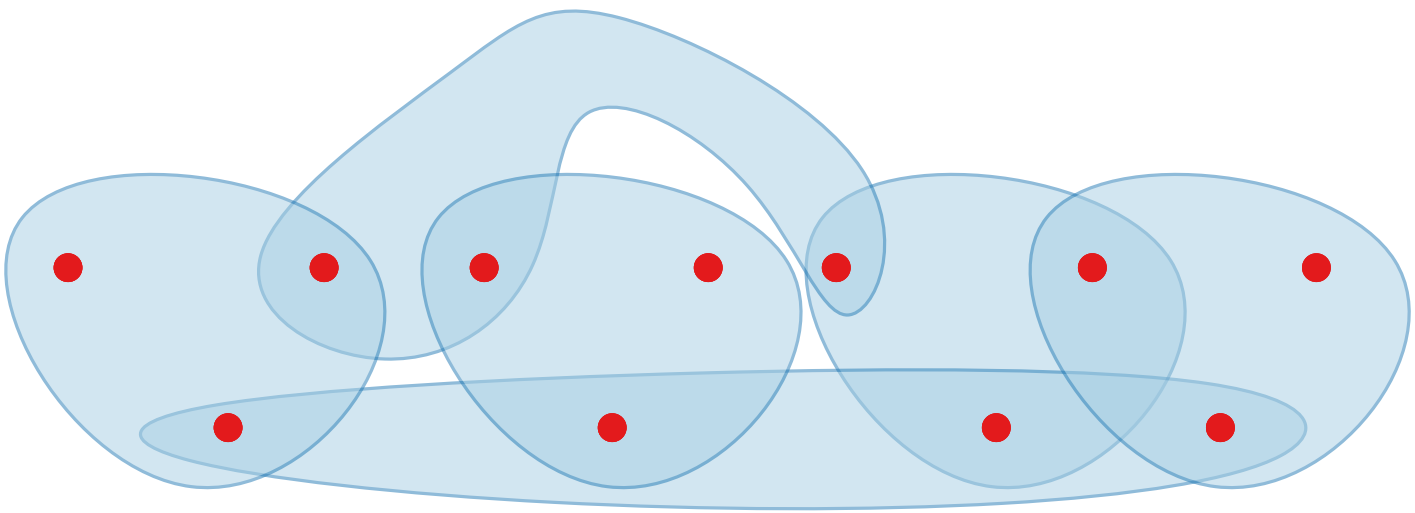
# SetCover as an ILP

Ground set $U$



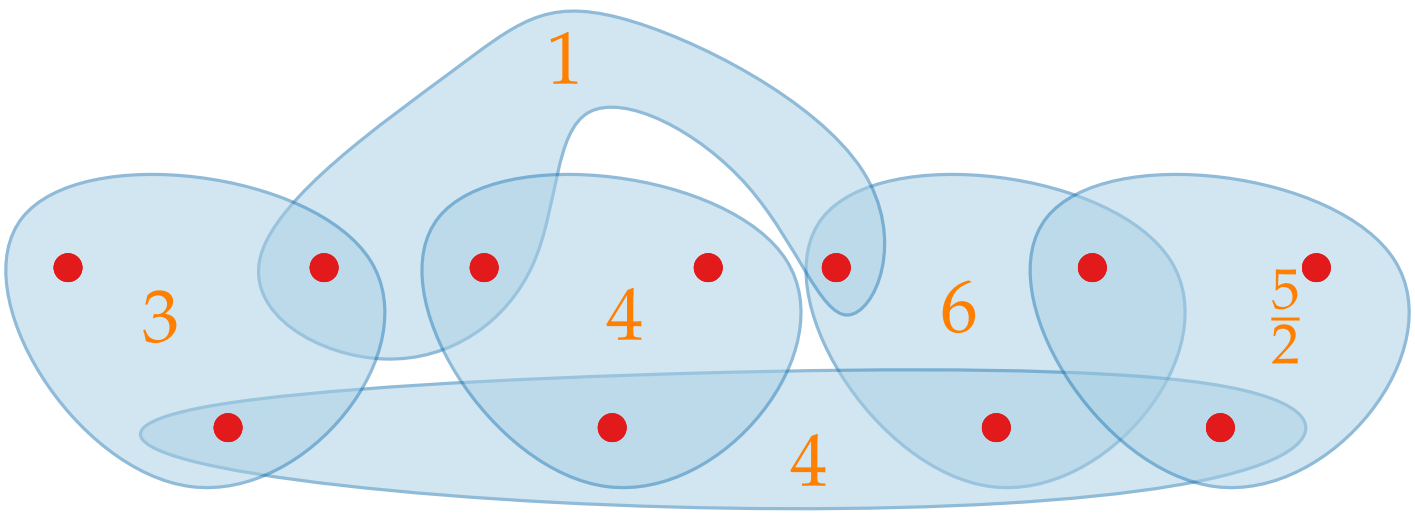
# SetCover as an ILP

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



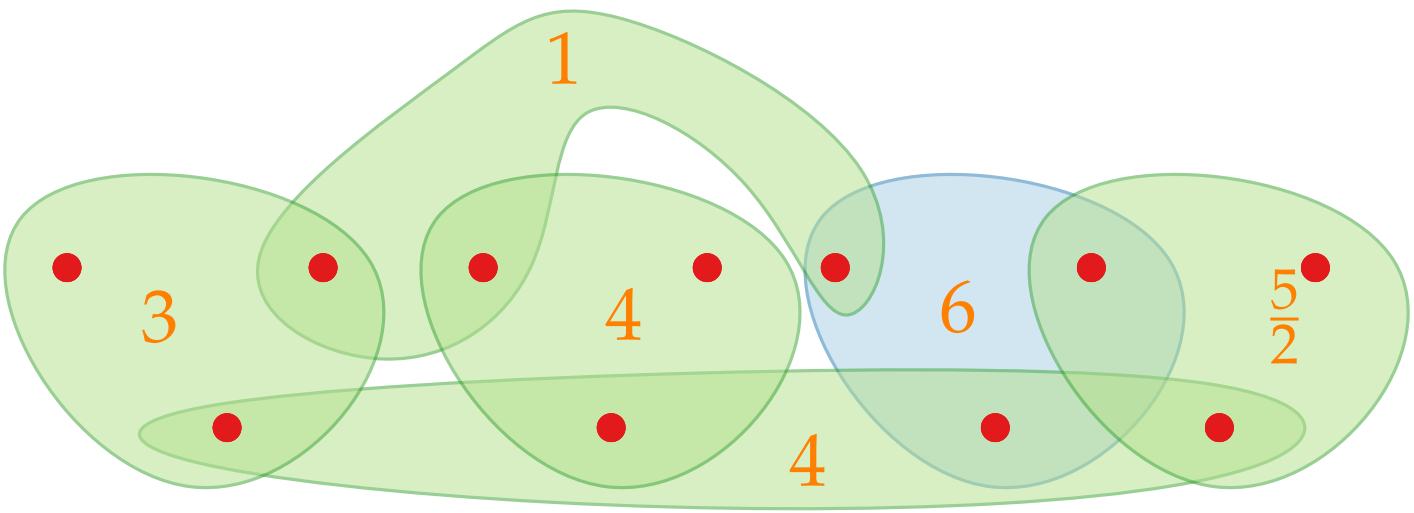
# SetCover as an ILP

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



# SetCover as an ILP

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



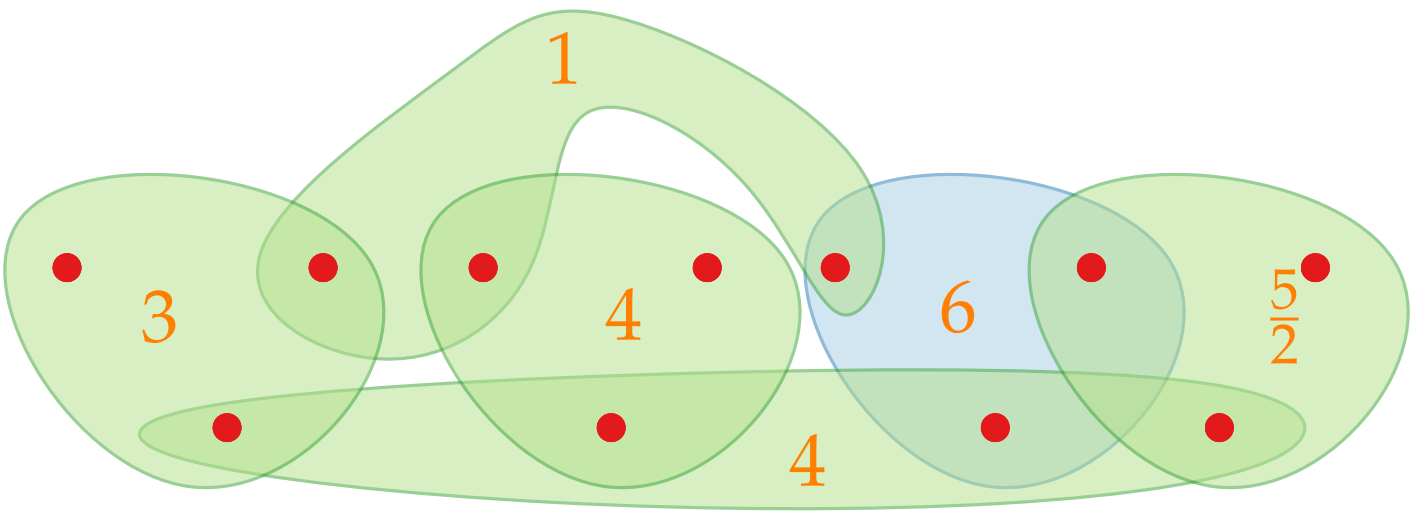

Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of $U$ with minimum cost.

# SetCover as an ILP

**minimize**

**subject to**

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




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

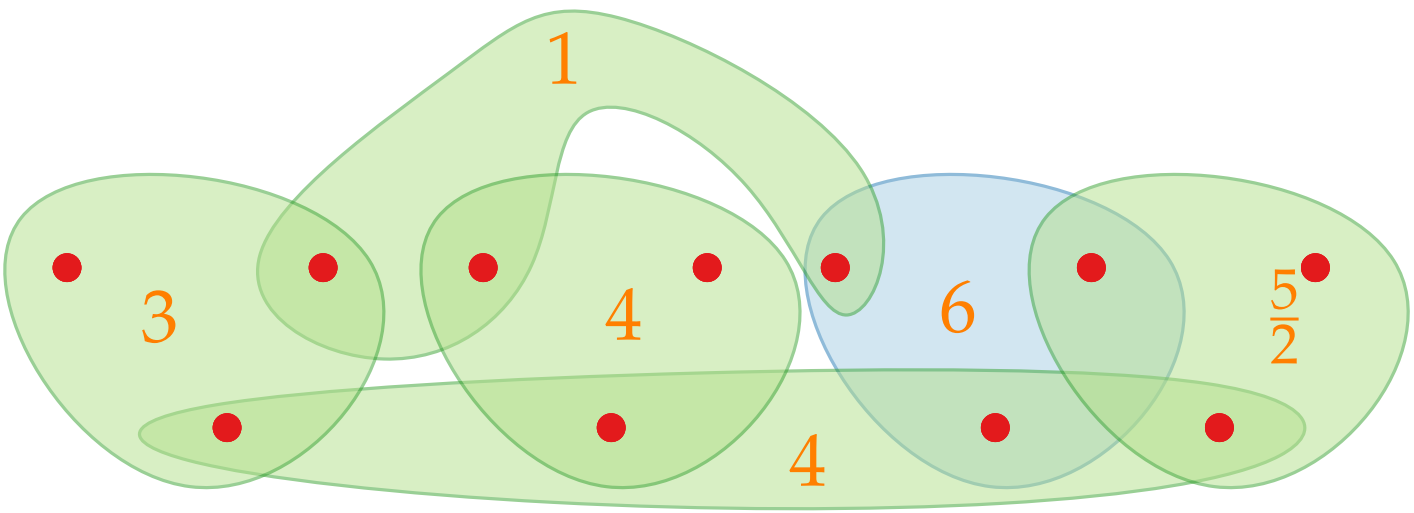
# SetCover as an ILP

**minimize**

**subject to**

$x_S$ $S \in \mathcal{S}$

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




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

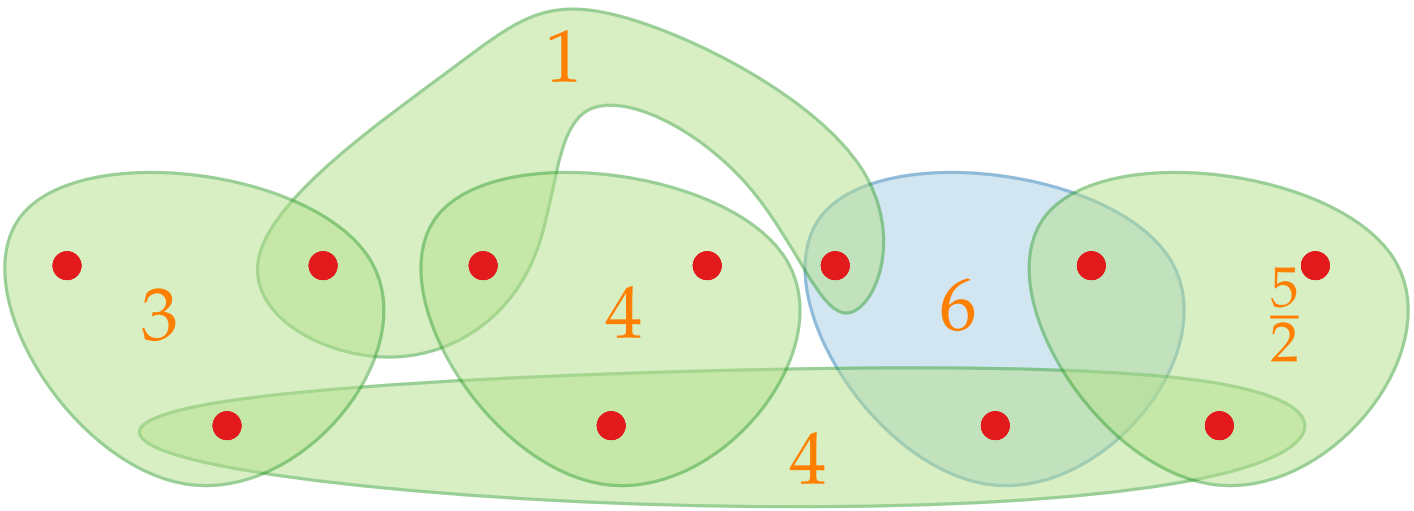
# SetCover as an ILP

**minimize**

**subject to**

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

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




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

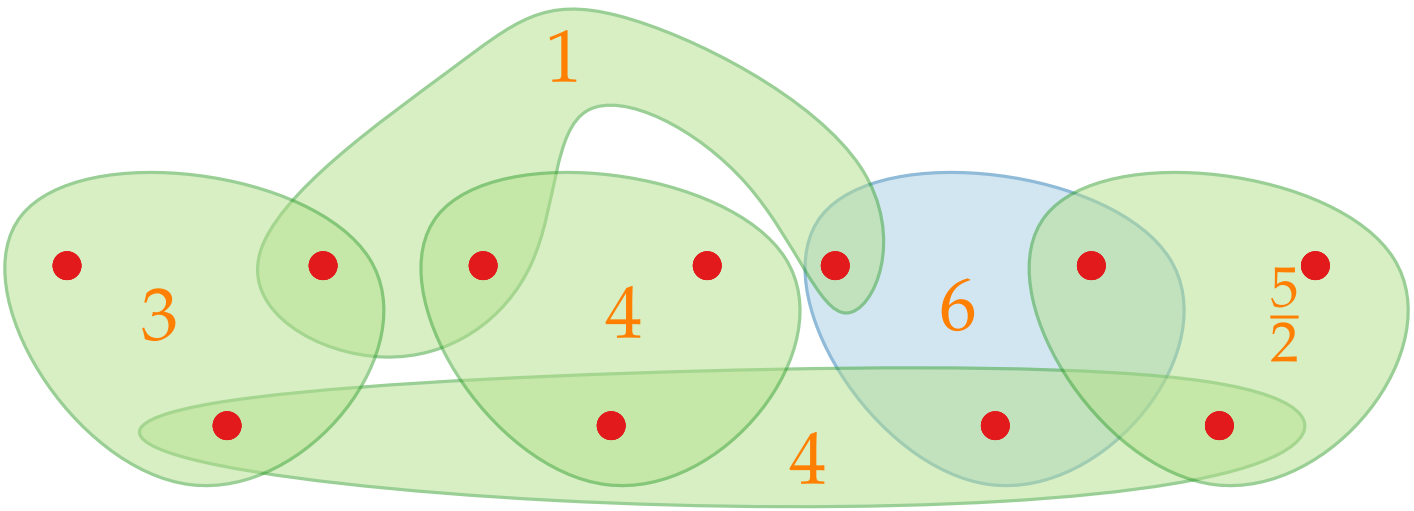
# SetCover as an ILP

**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 $c : \mathcal{S} \to \mathbb{Q}^+$

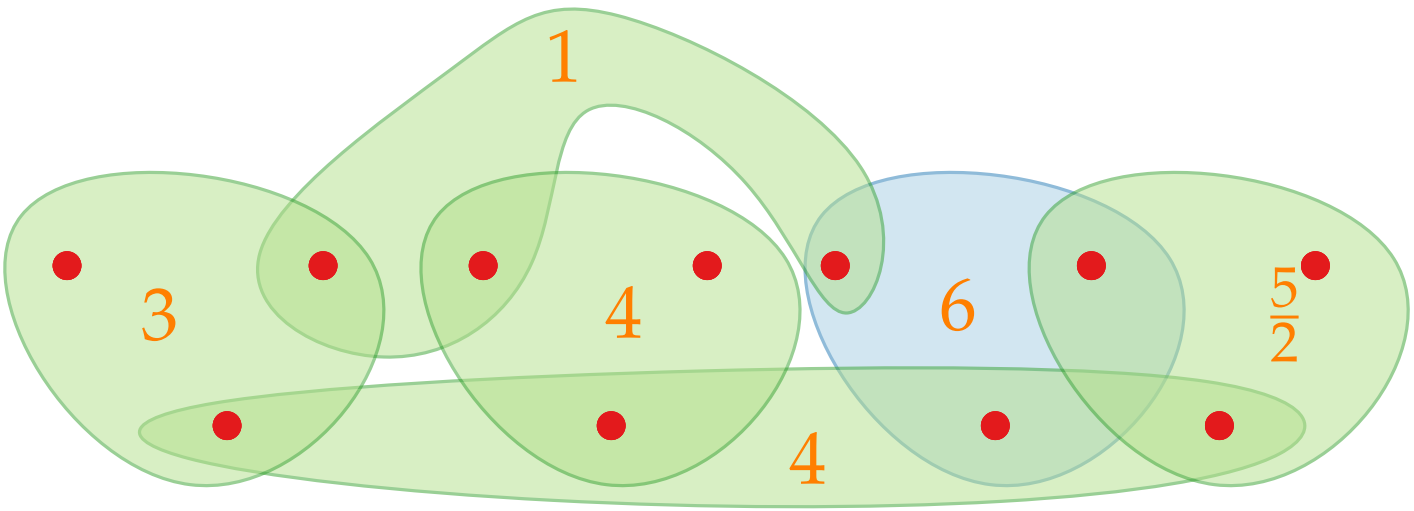



Find cover $\mathcal{S}' \subseteq \mathcal{S}$ of $U$ with minimum cost.

# SetCover as an ILP

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

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




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

# SetCover – LP-Relaxation

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

# SetCover – LP-Relaxation

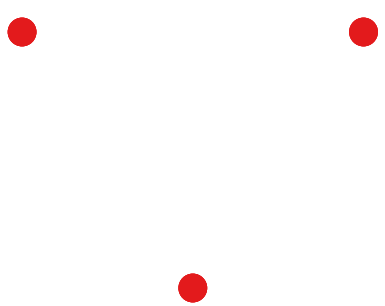
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S & \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 & u \in U \\ & x_S \geq 0 & S \in \mathcal{S} \end{aligned}$$

Optimal?

# SetCover – LP-Relaxation

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

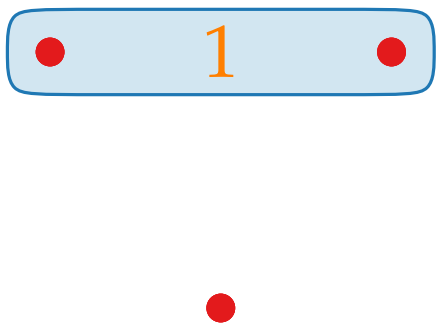
Optimal?



# SetCover – LP-Relaxation

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

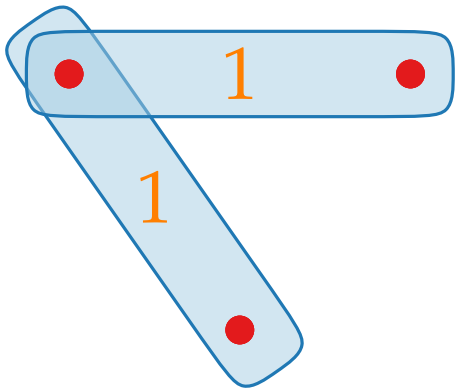
Optimal?



# SetCover – LP-Relaxation

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

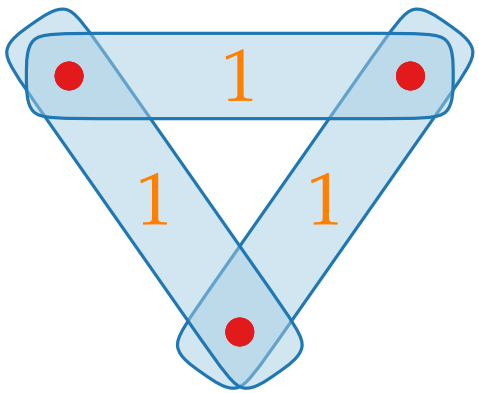
Optimal?



# SetCover – LP-Relaxation

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

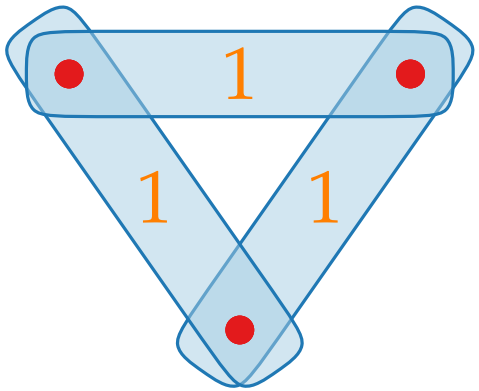
Optimal?

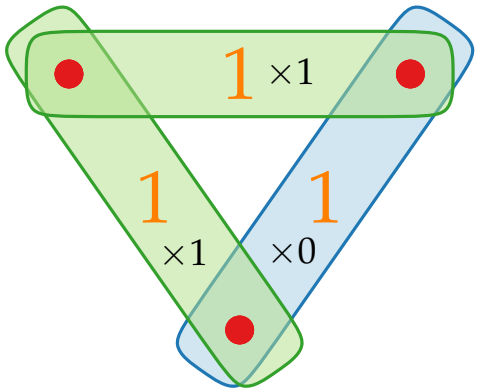


# SetCover – LP-Relaxation

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

Optimal?



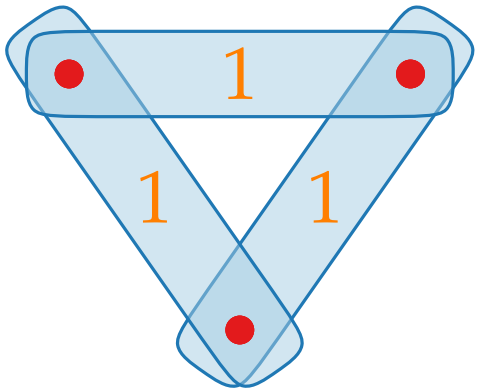



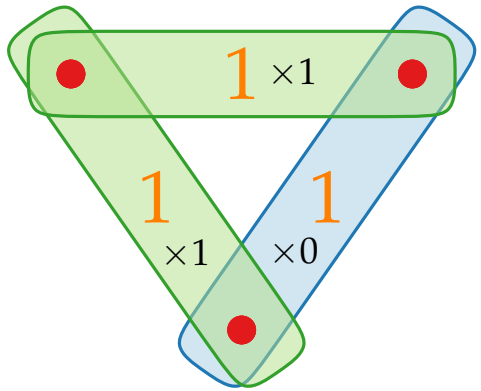

integer: 2

# SetCover – LP-Relaxation

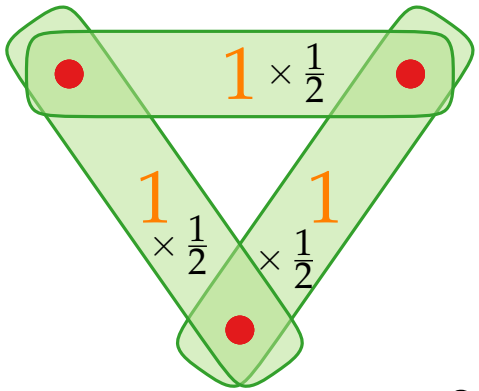
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

Optimal?




integer: 2




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

# SetCover – Dual LP

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

# 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

**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**

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

# SetCover – Dual LP

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

$$\begin{aligned} \textbf{maximize} \quad & \sum_{u \in U} y_u \\ \textbf{subject to} \quad & \\ & y_u \geq 0 \qquad u \in U \end{aligned}$$

# SetCover – Dual LP

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

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

# Approximation Algorithms

## Lecture 5:
## LP-based Approximation Algorithms for SetCover

### Part III:
### SetCover via LP-Rounding

Philipp Kindermann

Summer Semester 2020

# LP-Rounding: Approach I

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

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

# LP-Rounding: Approach I

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

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

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

# LP-Rounding: Approach I

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

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

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

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

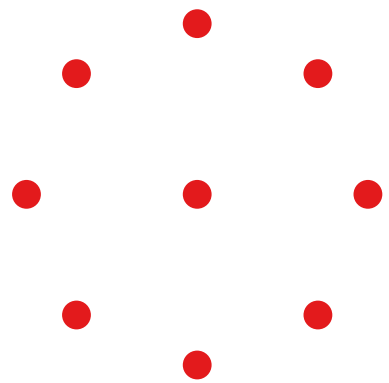
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

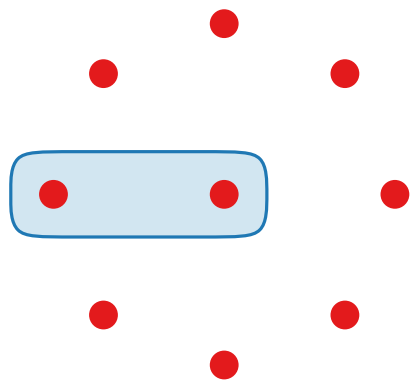
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

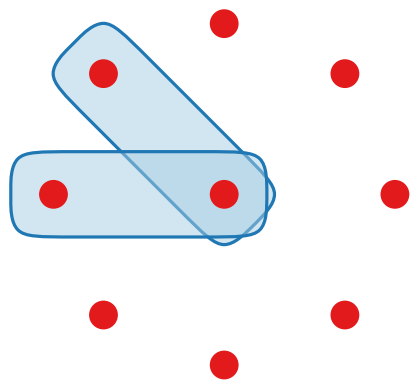
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

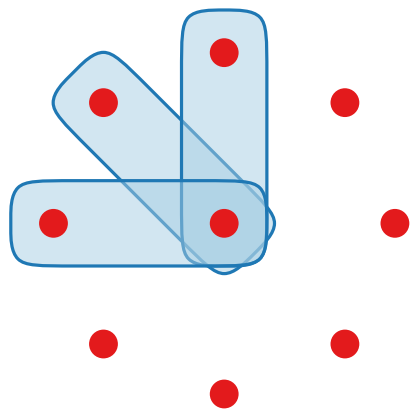
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

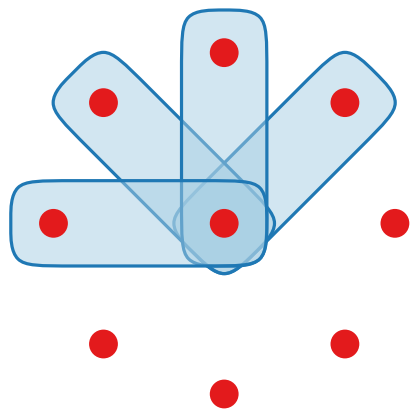
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

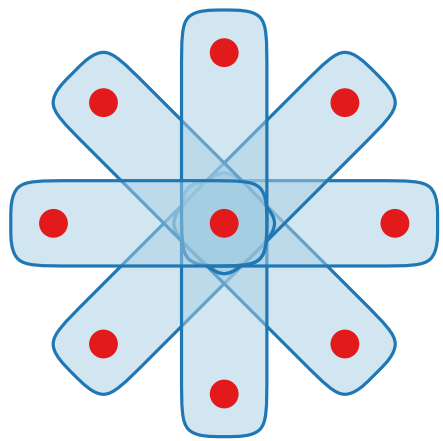
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

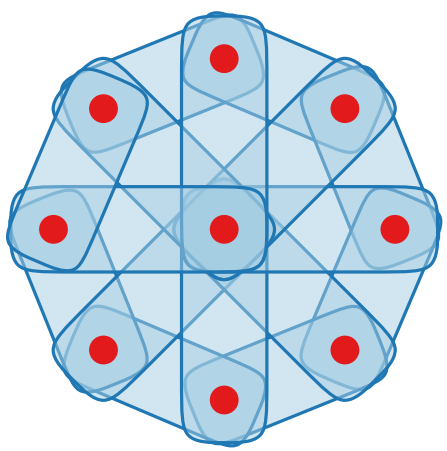
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

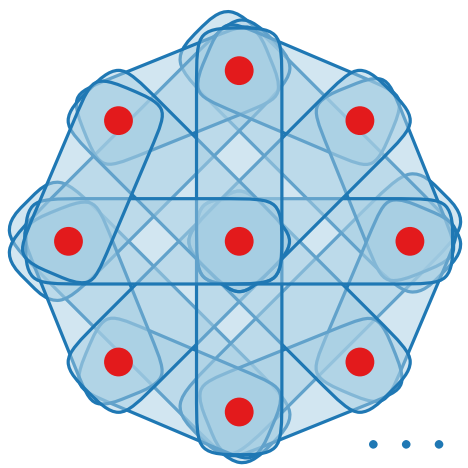
$$\begin{aligned} \textbf{minimize} \quad & \sum_{S \in \mathcal{S}} c_S x_S \\ \textbf{subject to} \quad & \sum_{S \ni u} x_S \geq 1 \qquad u \in U \\ & x_S \geq 0 \qquad S \in \mathcal{S} \end{aligned}$$

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

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

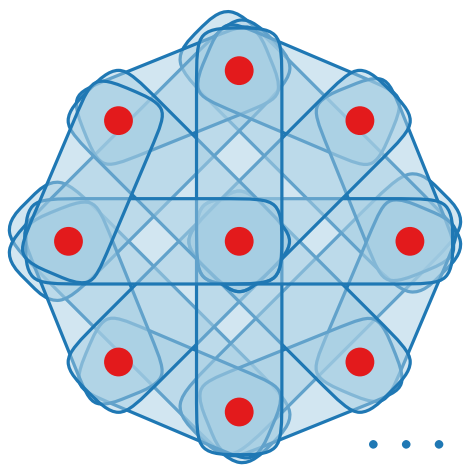
LP-Rounding-One($U, \mathcal{S}, c$)

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

Use frequency $h$



# LP-Rounding: Approach II

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

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

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

Let $h$ be the frequency of (number of sets containing) the most frequent element.

# LP-Rounding: Approach II

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

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

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

Let $h$ be the frequency of (number of sets containing) the most frequent element.

**Theorem.** LP-Rounding-Two is a factor-$h$-approximation algorithm for SetCover.

# Approximation Algorithms

## Lecture 5:
## LP-based Approximation Algorithms for SetCover

### Part IV:
### SetCover via Primal-Dual Schema

Philipp Kindermann

Summer Semester 2020

# Complementary Slackness

| | | | |
|---|---|---|---|
| **minimize** | $c^\top x$ | | |
| **subject to** | $Ax$ | $\geq$ | $b$ |
| | $x$ | $\geq$ | $0$ |

| | | | |
|---|---|---|---|
| **maximize** | $b^\top y$ | | |
| **subject to** | $A^\top y$ | $\leq$ | $c$ |
| | $y$ | $\geq$ | $0$ |

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

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

# Relaxing Complementary Slackness

| | | | |
|---|---|---|---|
| **minimize** | $c^\top x$ | | |
| **subject to** | $Ax$ | $\geq$ | $b$ |
| | $x$ | $\geq$ | $0$ |

| | | | |
|---|---|---|---|
| **maximize** | $b^\top y$ | | |
| **subject to** | $A^\top y$ | $\leq$ | $c$ |
| | $y$ | $\geq$ | $0$ |

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

**Dual CS**:
For each $i = 1, \ldots, m$: either $y_i = 0$ 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** | $c^\top x$ | | |
| **subject to** | $Ax$ | $\geq$ | $b$ |
| | $x$ | $\geq$ | $0$ |

| | | | |
|---|---|---|---|
| **maximize** | $b^\top y$ | | |
| **subject to** | $A^\top y$ | $\leq$ | $c$ |
| | $y$ | $\geq$ | $0$ |

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

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

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

**Dual CS**:

For each $i = 1, \ldots, m$: either $y_i = 0$ 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** | $c^\top x$ | | |
| **subject to** | $Ax$ | $\geq$ | $b$ |
| | $x$ | $\geq$ | $0$ |

| | | | |
|---|---|---|---|
| **maximize** | $b^\top y$ | | |
| **subject to** | $A^\top y$ | $\leq$ | $c$ |
| | $y$ | $\geq$ | $0$ |

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

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

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

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

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

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

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

# 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 $j = 1, \ldots, n$: either $x_j = 0$ or ~~$\sum_{i=1}^{m} a_{ij} y_i = c_j$~~

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

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

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

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

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

# 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 obj. 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 obj. 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 obj. 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 obj. 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 relaxed CS condition provide an approximation ratio.

# Relaxed CS for SetCover

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

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

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

# 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:**

# 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**: $x_S \neq 0 \Rightarrow$

# Relaxed CS for SetCover

| | | | | | |
|---|---|---|---|---|---|
| **minimize** | $\sum_{S \in \mathcal{S}} c_S x_S$ | | **maximize** | $\sum_{u \in U} y_u$ | |
| **subject to** | $\sum_{S \ni u} x_S \geq 1$ | $u \in U$ | **subject to** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ |
| | $x_S \geq 0$ | $S \in \mathcal{S}$ | | $y_u \geq 0$ | $u \in U$ |

**(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** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ | |
| | $y_u \geq 0$ | $u \in U$ | |

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

critical set

only chooses critical sets

# Relaxed CS for SetCover

| | | | | | |
|---|---|---|---|---|---|
| **minimize** | $\sum_{S \in \mathcal{S}} c_S x_S$ | | **maximize** | $\sum_{u \in U} y_u$ | |
| **subject to** | $\sum_{S \ni u} x_S \geq 1$ | $u \in U$ | **subject to** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ |
| | $x_S \geq 0$ | $S \in \mathcal{S}$ | | $y_u \geq 0$ | $u \in U$ |

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

critical set

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** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ | |
| | $y_u \geq 0$ | $u \in U$ | |

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

critical set

only chooses critical sets

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

# Relaxed CS for SetCover

| | | | | | |
|---|---|---|---|---|---|
| **minimize** | $\sum_{S \in \mathcal{S}} c_S x_S$ | | **maximize** | $\sum_{u \in U} y_u$ | |
| **subject to** | $\sum_{S \ni u} x_S \geq 1$ | $u \in U$ | **subject to** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ |
| | $x_S \geq 0$ | $S \in \mathcal{S}$ | | $y_u \geq 0$ | $u \in U$ |

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:** $y_u \neq 0 \Rightarrow 1 \leq \sum_{S \ni u} x_S \leq h$ .

# Relaxed CS for SetCover

| | | | | | |
|---|---|---|---|---|---|
| **minimize** | $\sum_{S \in \mathcal{S}} c_S x_S$ | | **maximize** | $\sum_{u \in U} y_u$ | |
| **subject to** | $\sum_{S \ni u} x_S \geq 1$ | $u \in U$ | **subject to** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ |
| | $x_S \geq 0$ | $S \in \mathcal{S}$ | | $y_u \geq 0$ | $u \in U$ |

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:** $y_u \neq 0 \Rightarrow 1 \leq \sum_{S \ni u} x_S \leq h \cdot 1$

# Relaxed CS for SetCover

| | | | | | |
|---|---|---|---|---|---|
| **minimize** | $\sum_{S \in \mathcal{S}} c_S x_S$ | | **maximize** | $\sum_{u \in U} y_u$ | |
| **subject to** | $\sum_{S \ni u} x_S \geq 1$ | $u \in U$ | **subject to** | $\sum_{u \in S} y_u \leq c_S$ | $S \in \mathcal{S}$ |
| | $x_S \geq 0$ | $S \in \mathcal{S}$ | | $y_u \geq 0$ | $u \in U$ |

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:** $y_u \neq 0 \Rightarrow 1 \leq \sum_{S \ni u} x_S \leq h \cdot 1$

# 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 ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

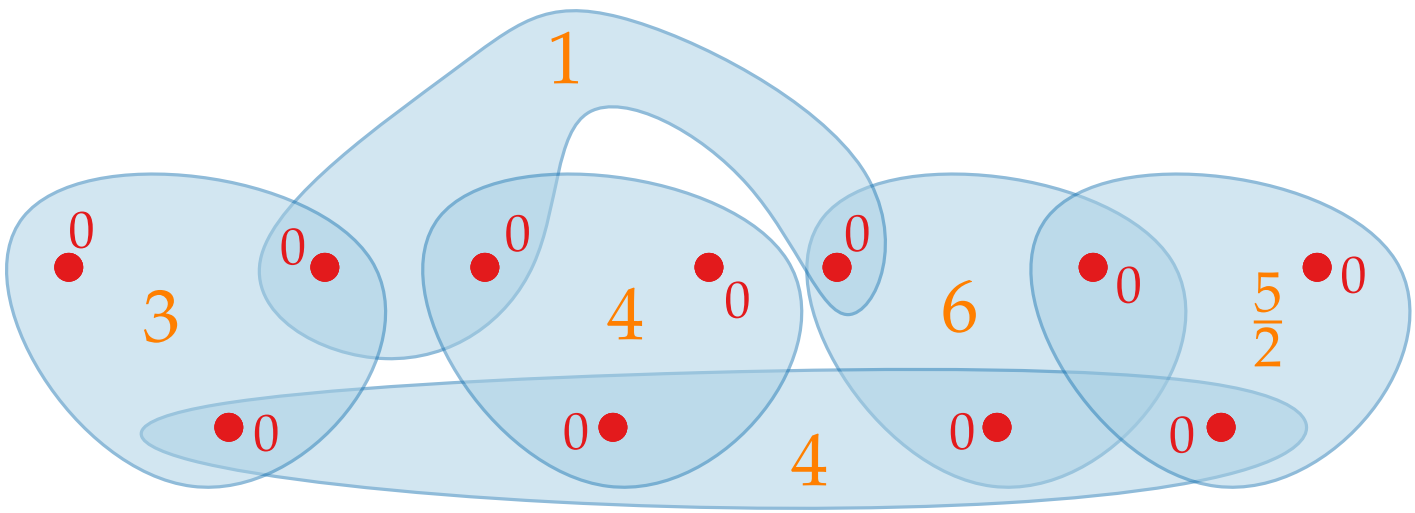
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

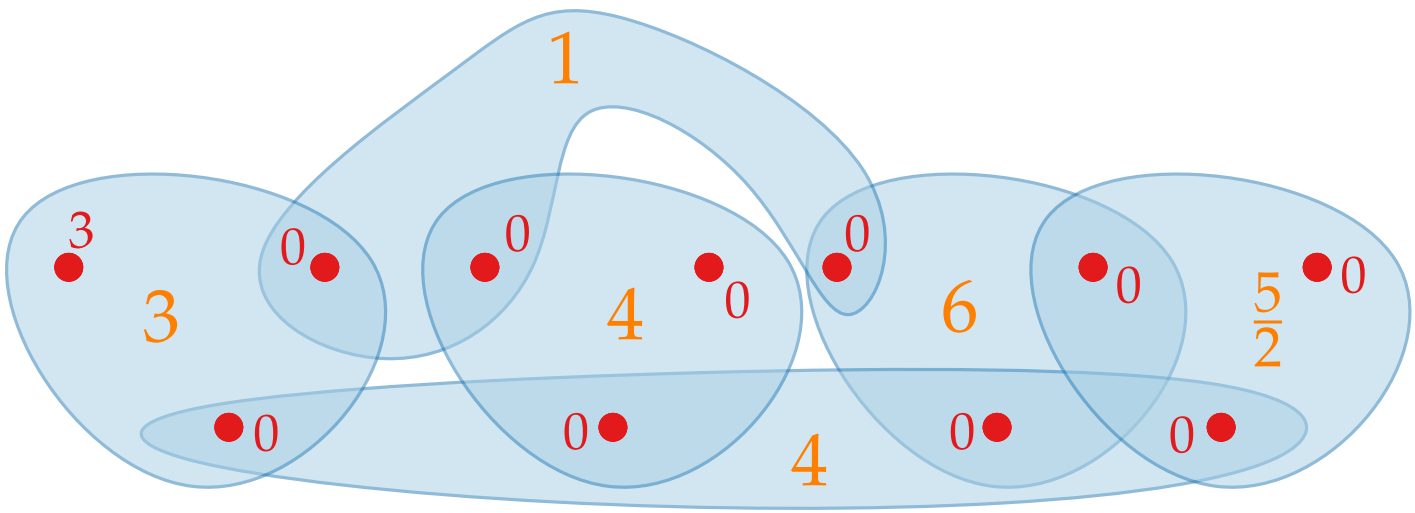
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

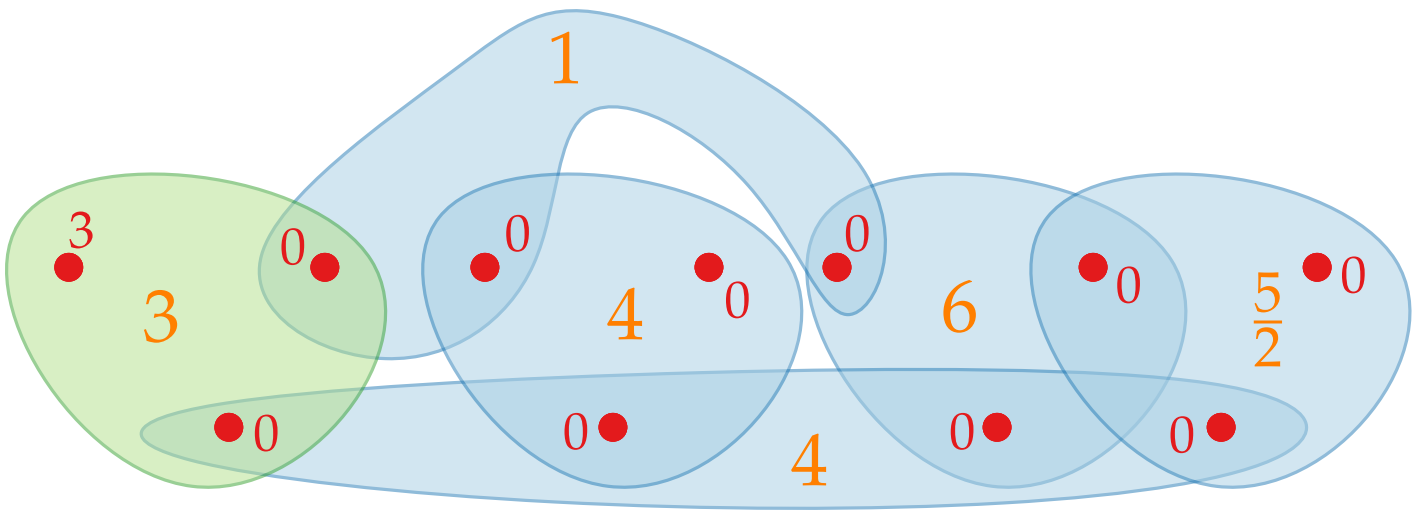
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

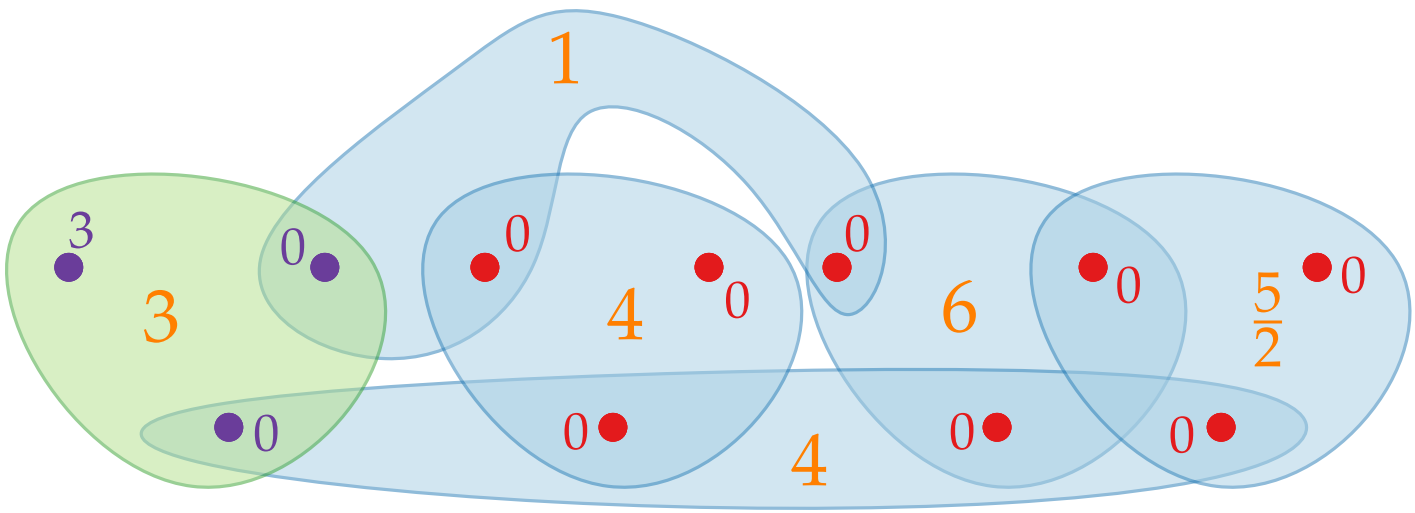
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

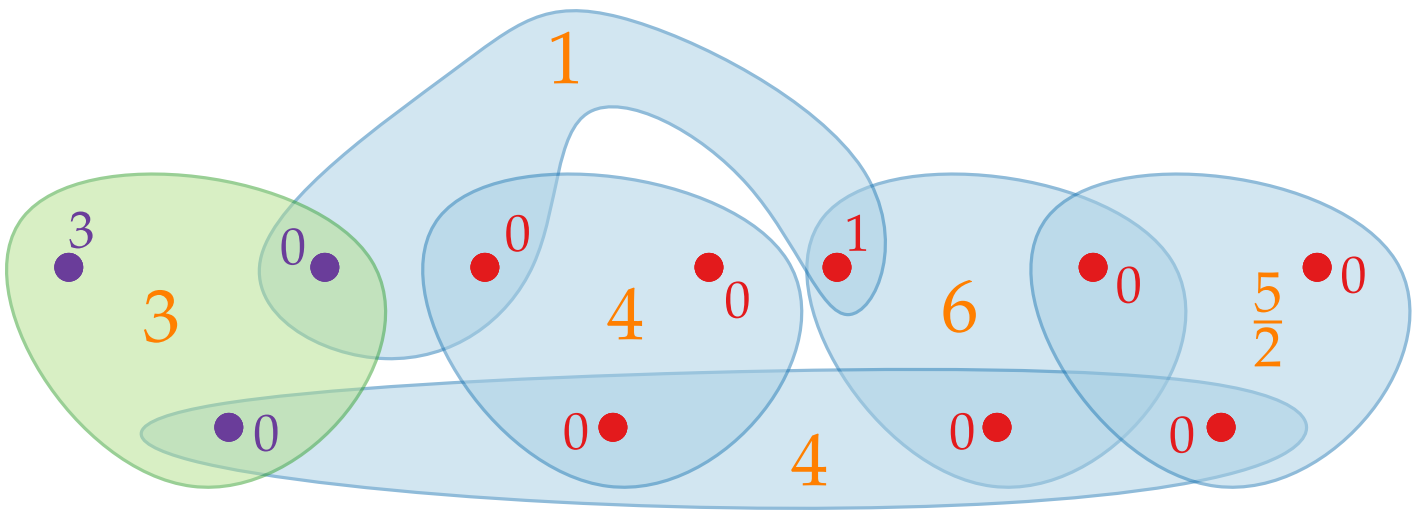
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

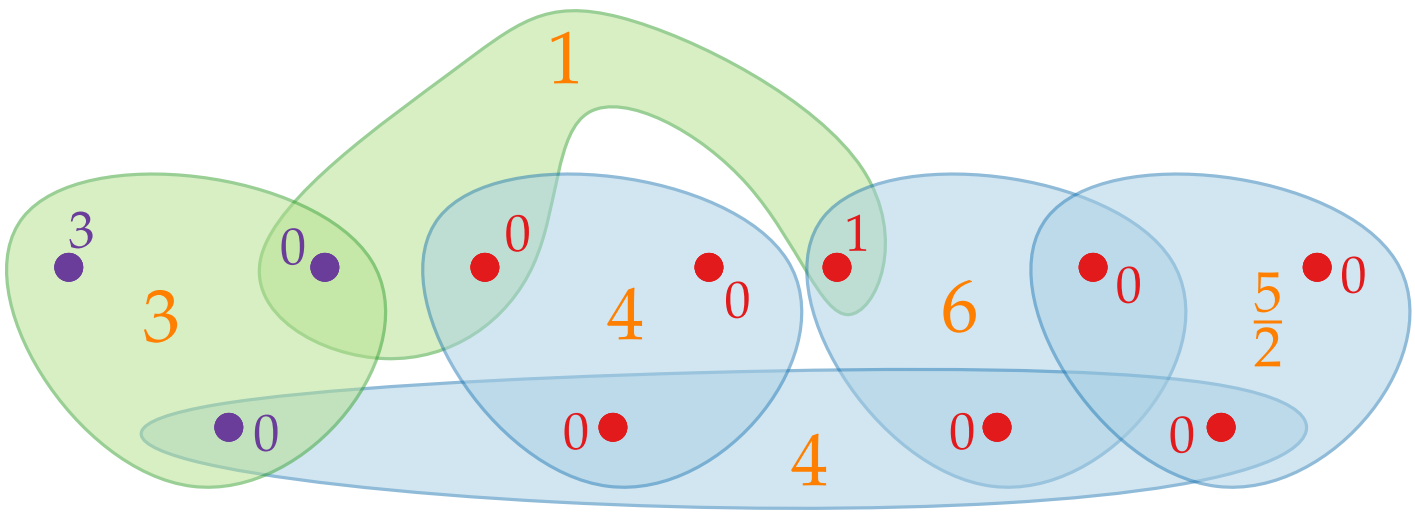
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

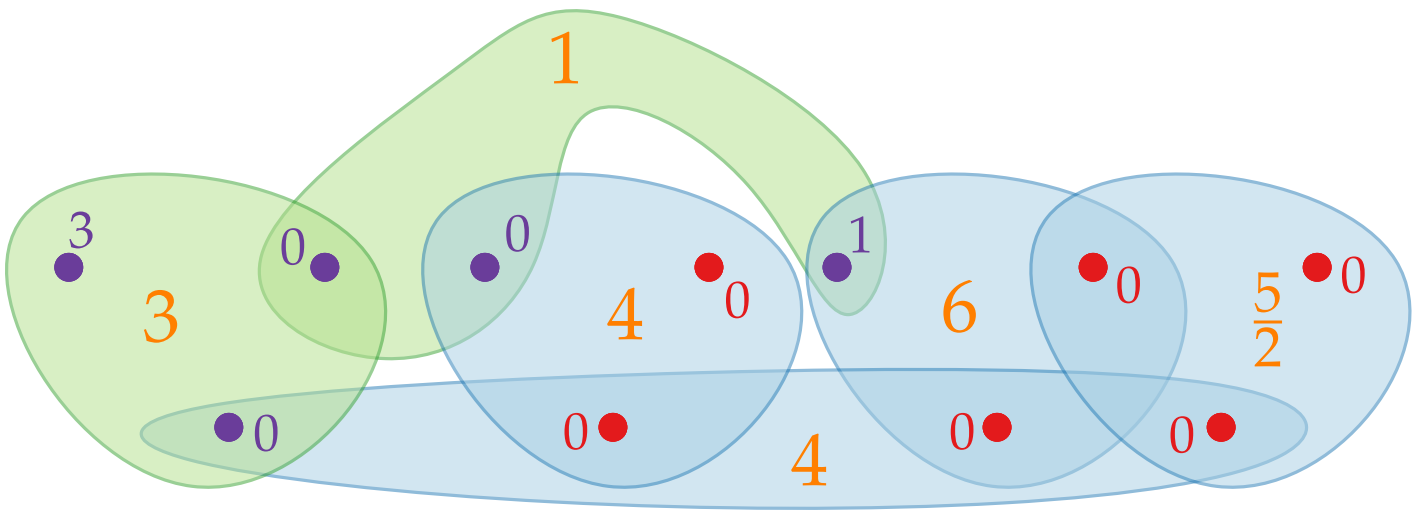
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

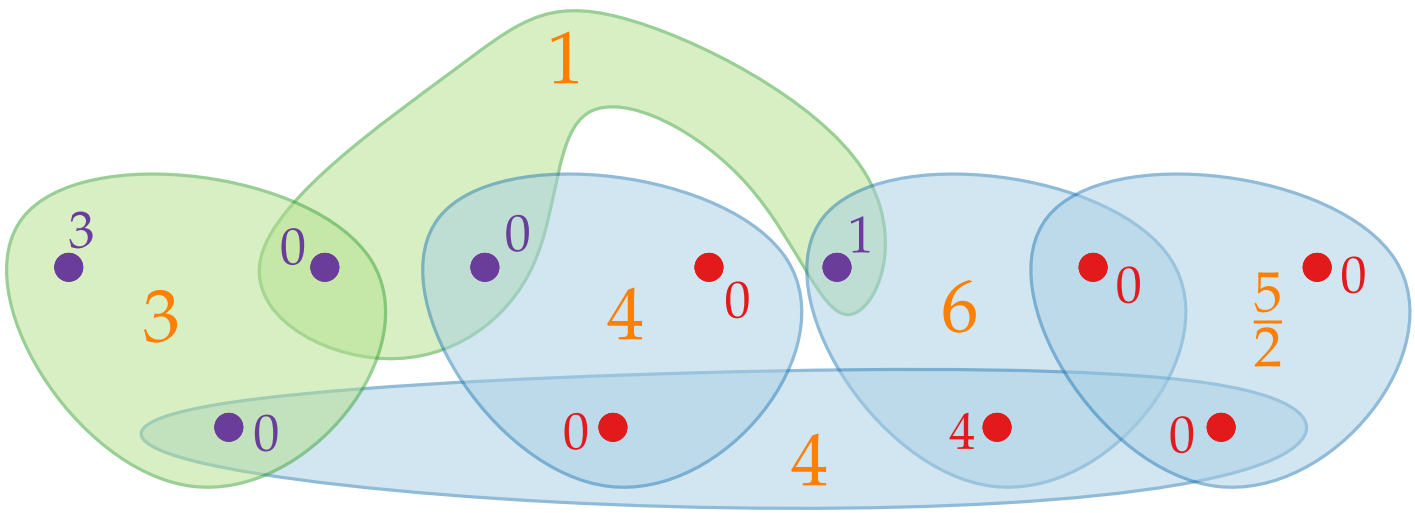
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

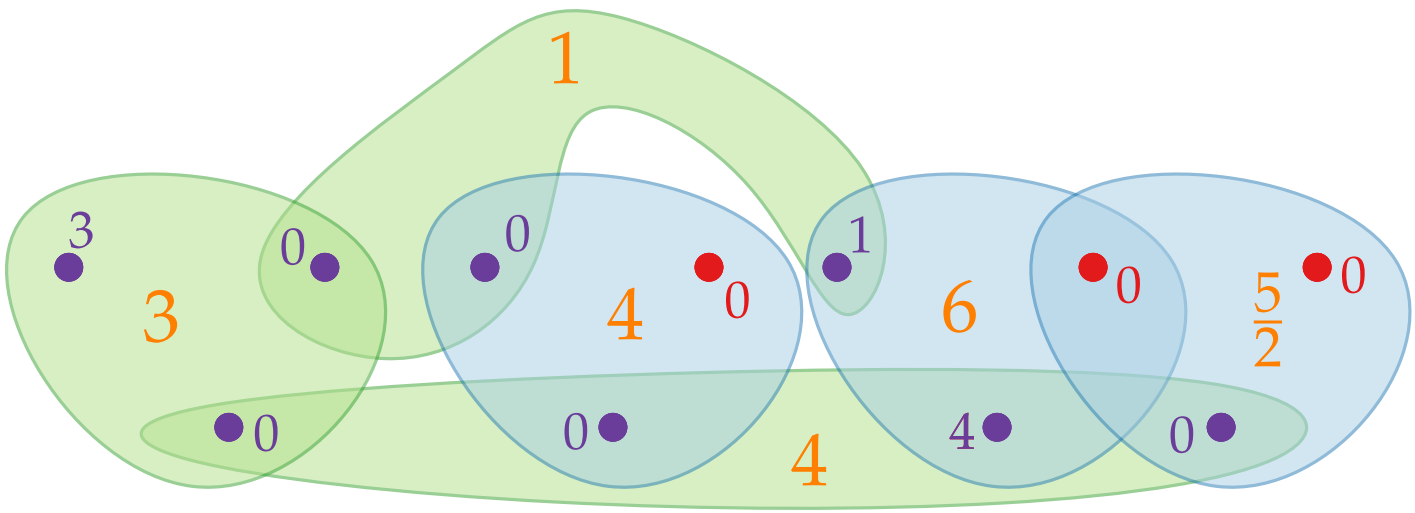
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

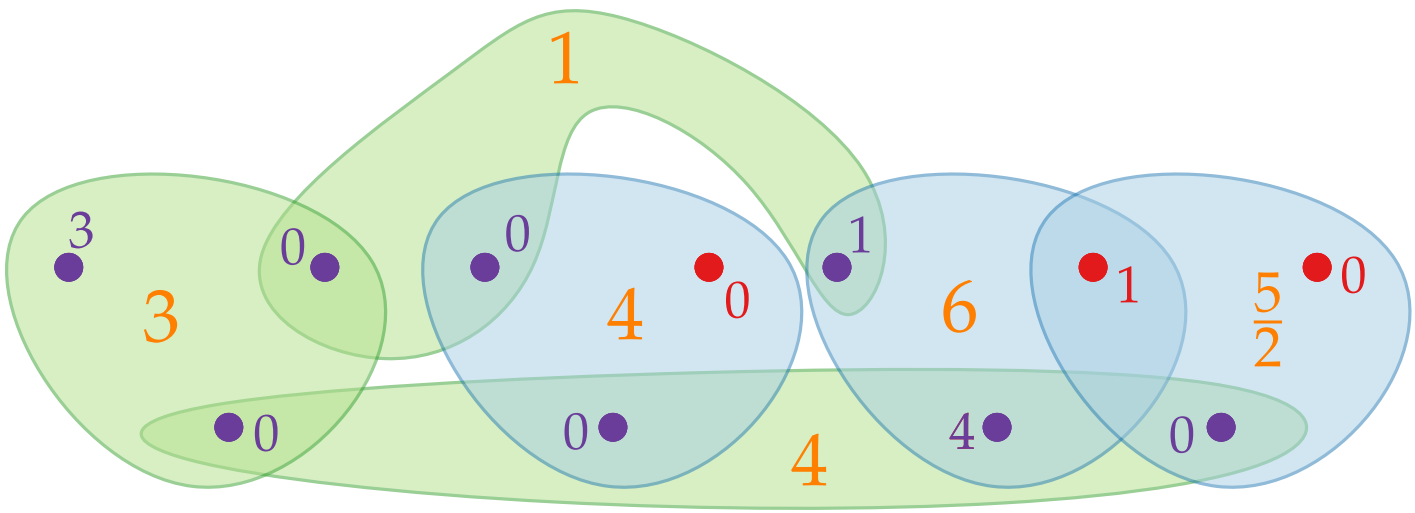
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

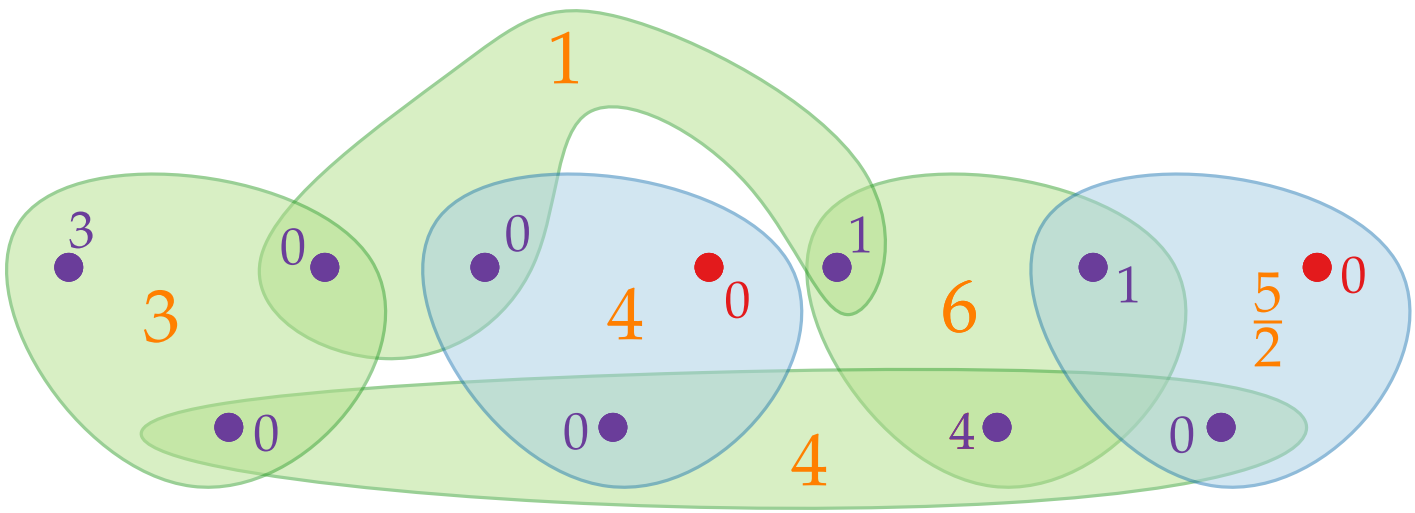
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

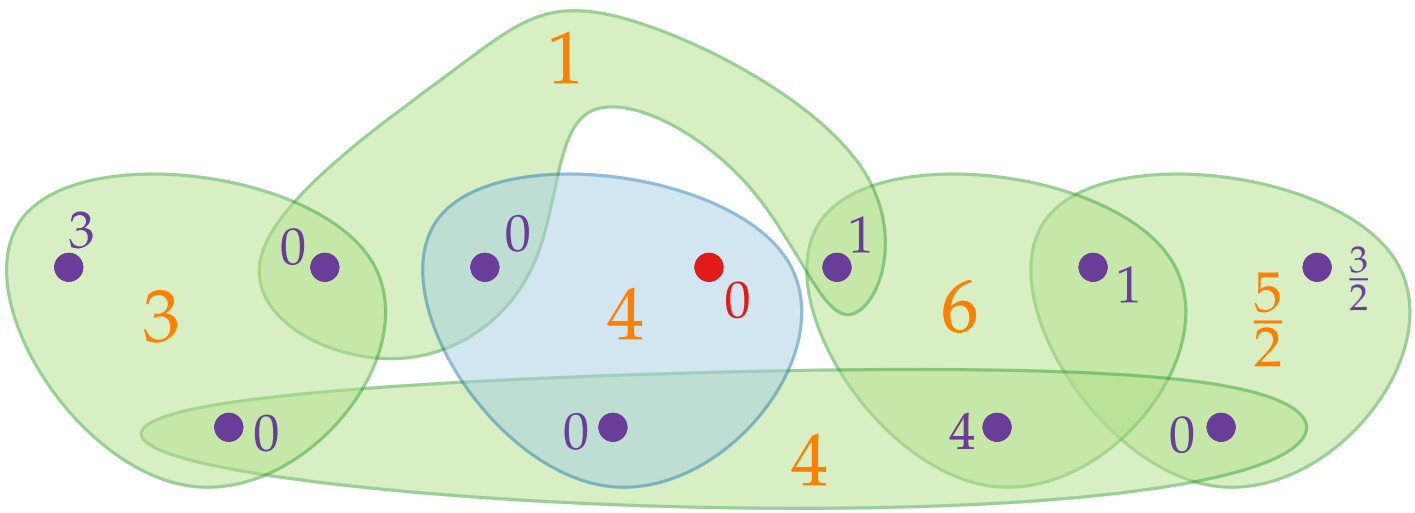
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 $u$.

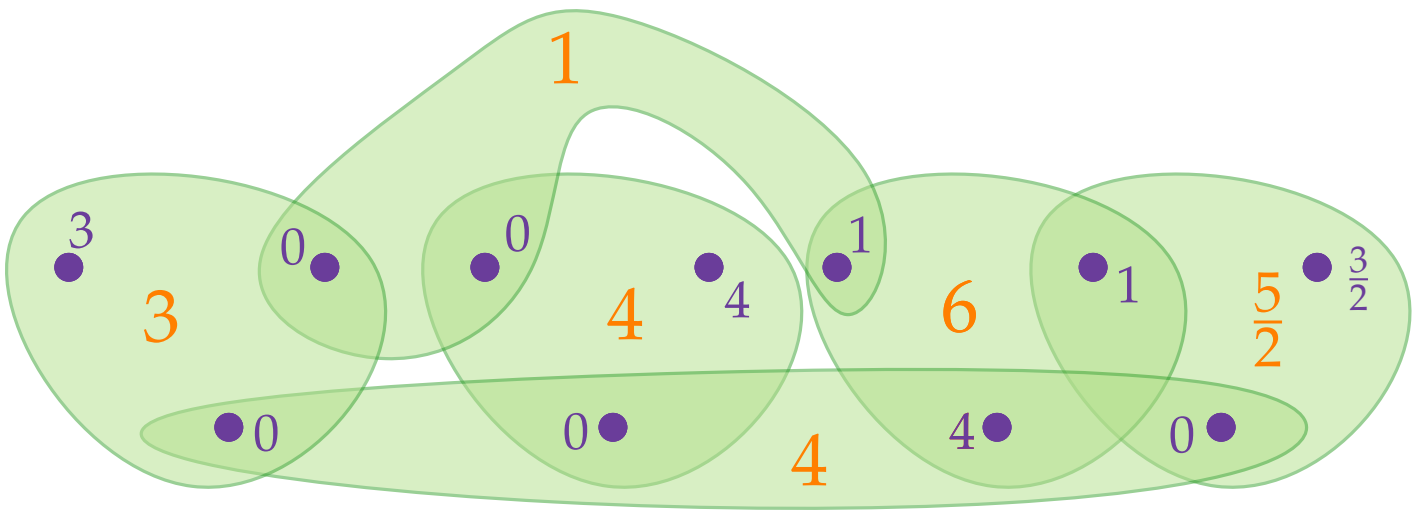
  Increase $y_u$ until a set $S$ is critical ($\sum_{u' \in S} y_{u'} = c_S$).

  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 ($\sum_{u' \in S} y_{u'} = c_S$).

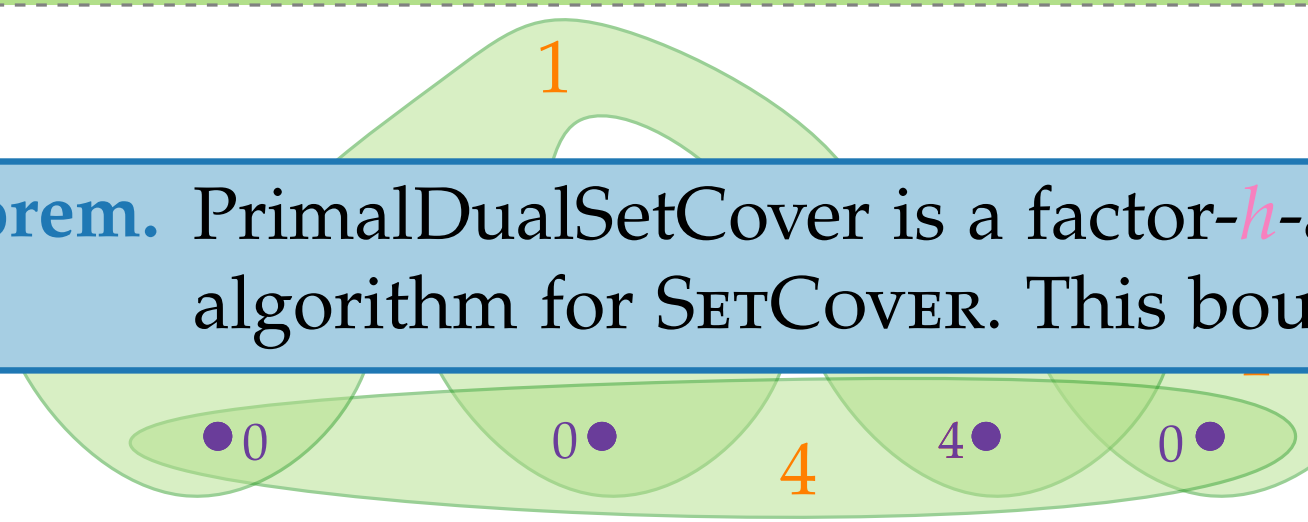
  Select all critical sets and update $x$.

  Mark all elements in these sets as covered.

**until** all elements are covered.

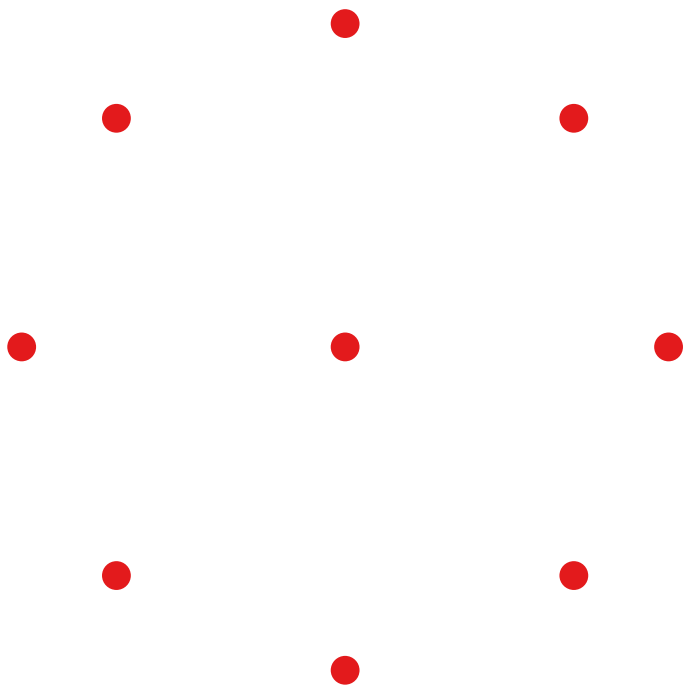
**return** $x$

**Theorem.** PrimalDualSetCover is a factor-$h$-approximation algorithm for SetCover. This bound is tight.

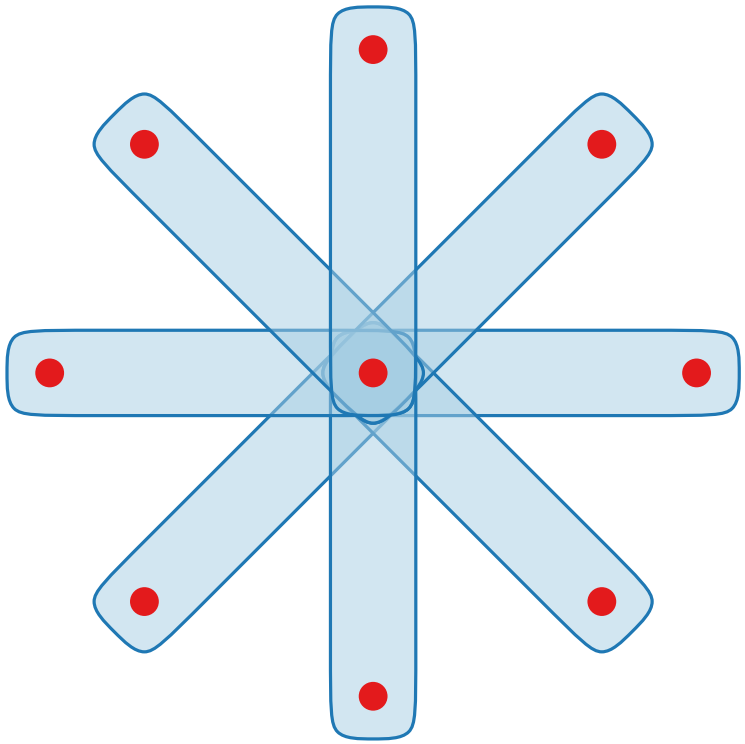


# Tight Example

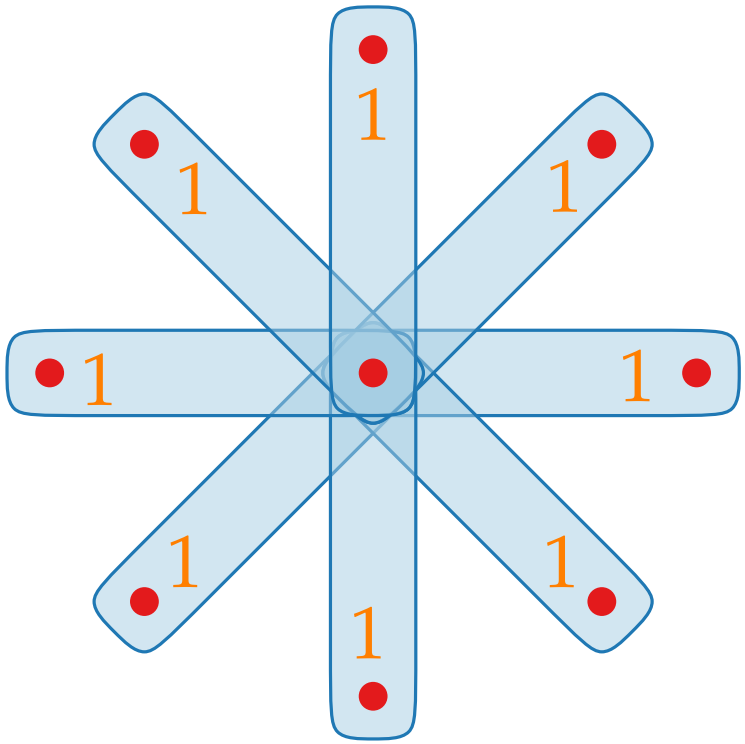
# Tight Example



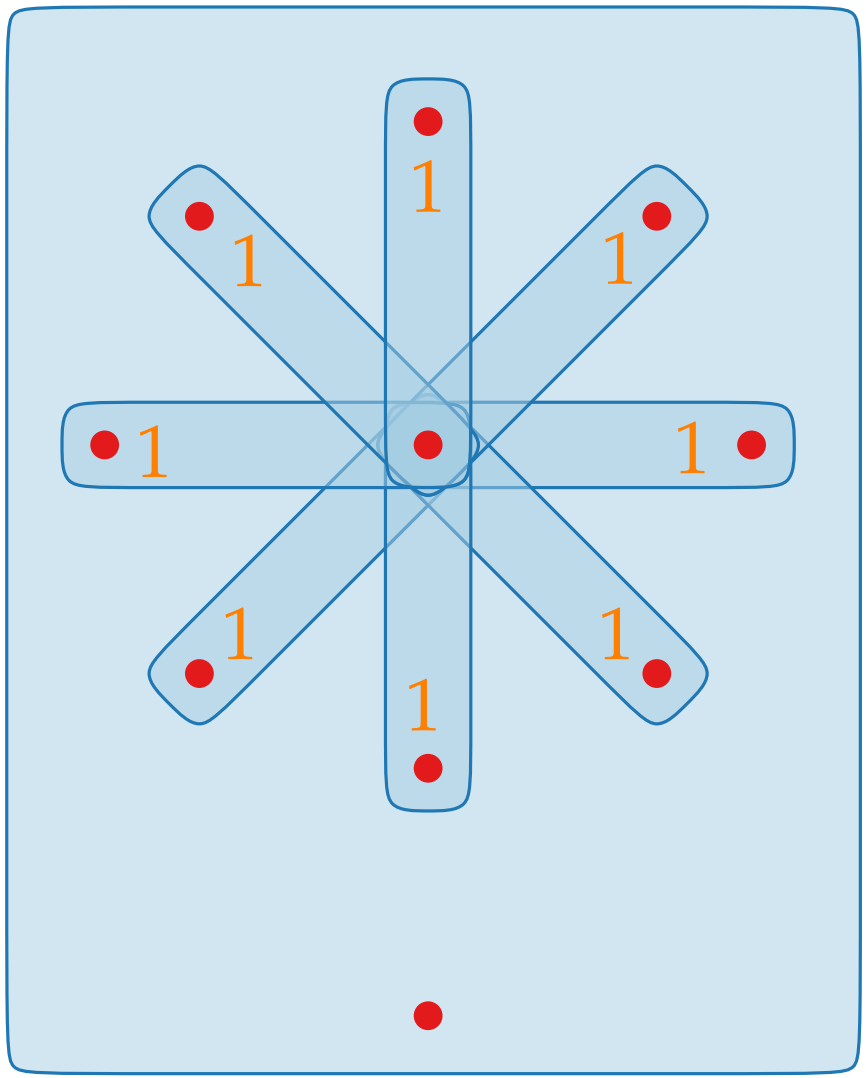
# Tight Example



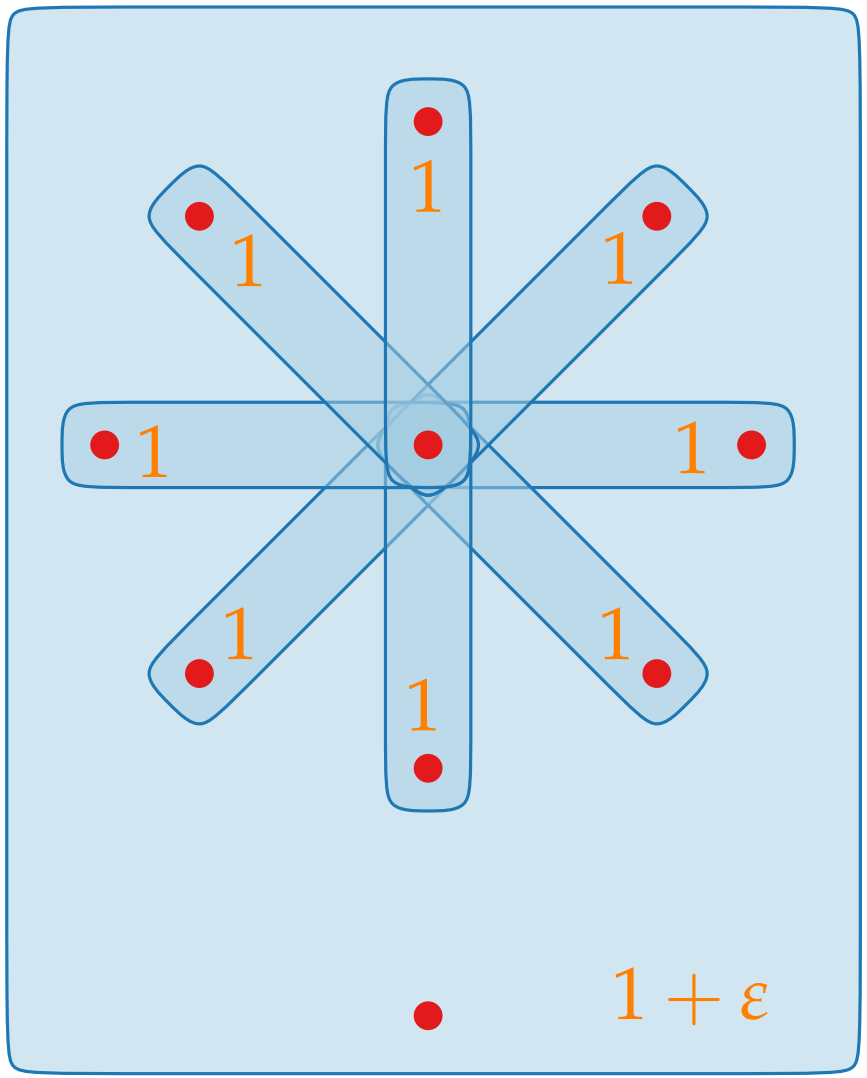
# Tight Example



# Tight Example



# Tight Example



# Approximation Algorithms

## Lecture 5:
## LP-based Approximation Algorithms for SetCover

### Part V:
### SetCover via Dual Fitting

Philipp Kindermann

Summer Semester 2020

# Dual Fitting for SetCover

Combinatorial (greedy) algorithm (see Lecture 02):

```
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 02):

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

# Dual Fitting for SetCover

Combinatorial (greedy) algorithm (see Lecture 02):

```
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 $u \in U$, $\text{price}(u)$ is a dual variable

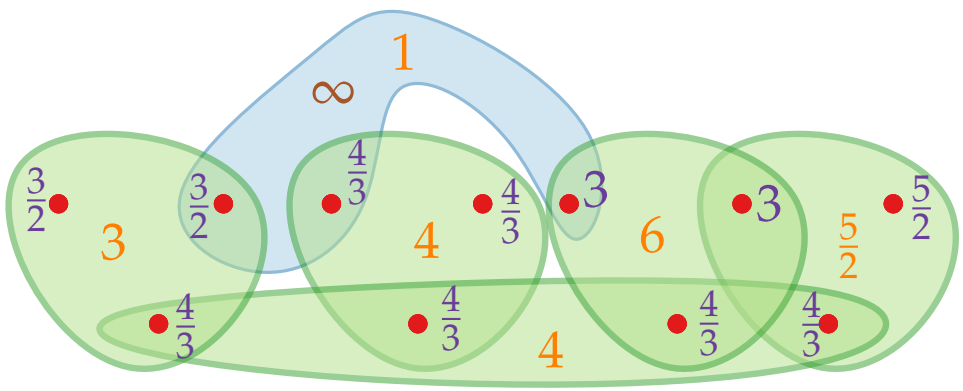
# New: LP-based Analysis

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

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

# New: LP-based Analysis

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

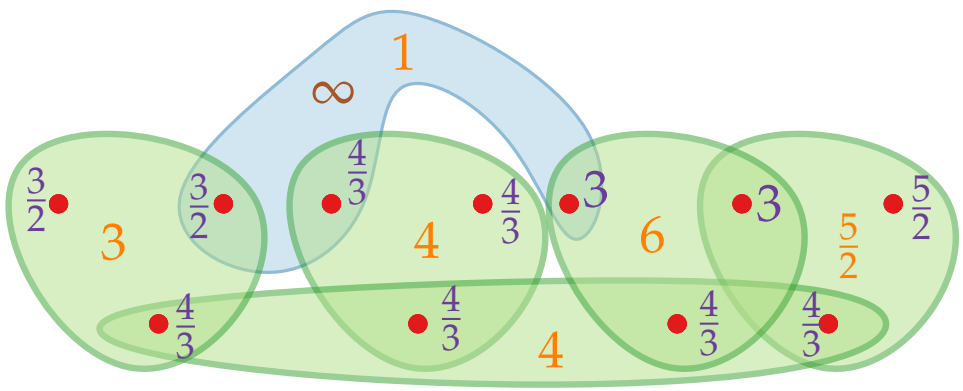



$$\begin{aligned} \textbf{maximize} \quad & \sum_{u \in U} y_u \\ \textbf{subject to} \quad & \sum_{u \in S} y_u \leq c_S \qquad S \in \mathcal{S} \\ & y_u \geq 0 \qquad u \in U \end{aligned}$$

# 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.

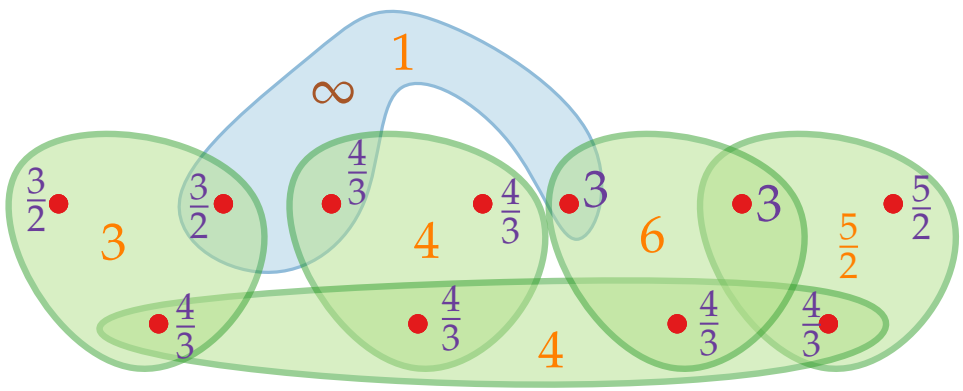



$$\begin{aligned} \textbf{maximize} \quad & \sum_{u \in U} y_u \\ \textbf{subject to} \quad & \sum_{u \in S} y_u \leq c_S \qquad S \in \mathcal{S} \\ & y_u \geq 0 \qquad u \in U \end{aligned}$$

# 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.

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




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

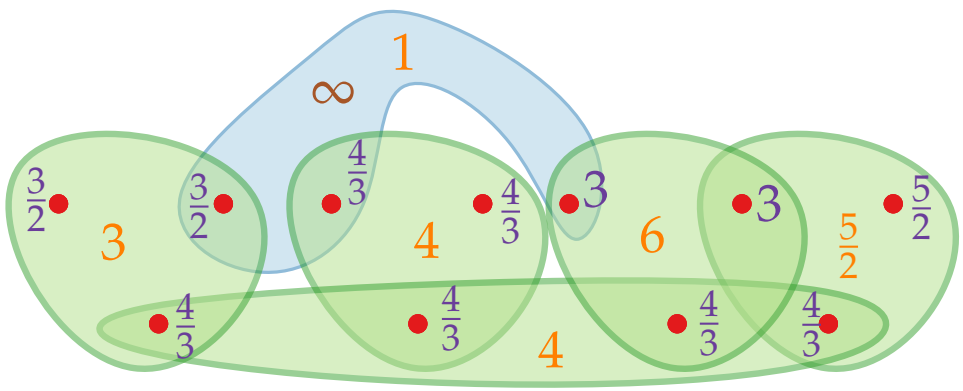
# New: LP-based Analysis

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

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

*Dual-Fitting-Trick:*

Scale duale variables such that no set is overpacked.




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

# New: LP-based Analysis

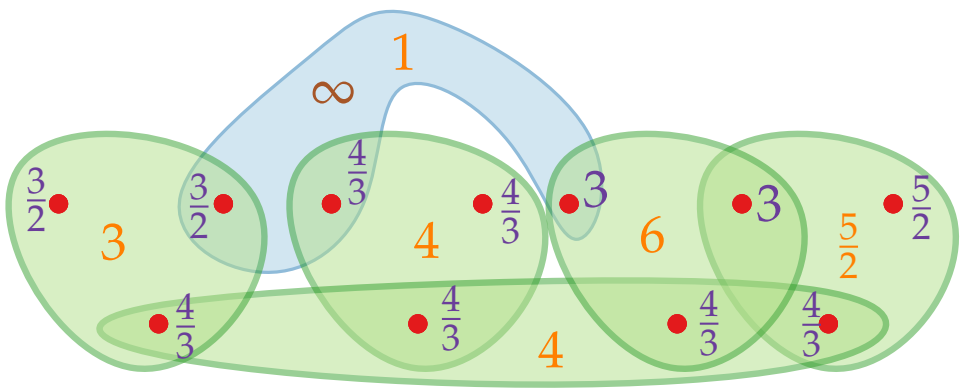
**Observation.** For each $u \in U$, price$(u)$ is a dual variable
But this dual solution is in general not feasible.

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

*Dual-Fitting-Trick:*

Scale duale variables such that no set is overpacked.

Take $y_u =$




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

# New: LP-based Analysis

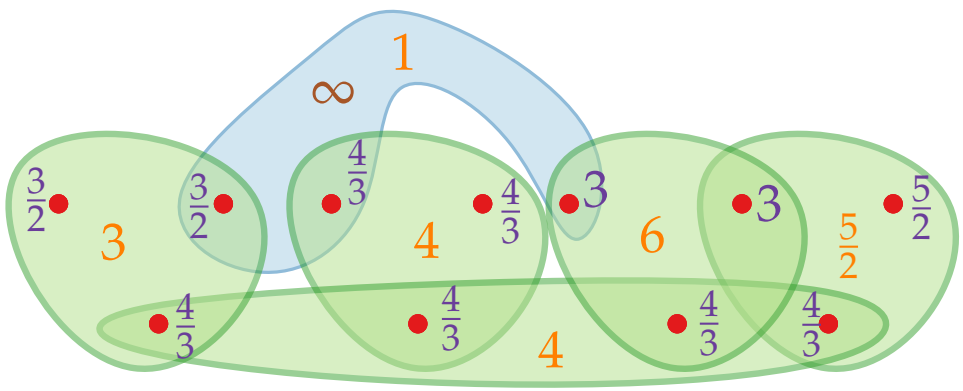
**Observation.** For each $u \in U$, price$(u)$ is a dual variable
But this dual solution is in general not feasible.

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

*Dual-Fitting-Trick:*

Scale duale variables such that no set is overpacked.

Take $y_u = \text{price}(u)/$




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

# New: LP-based Analysis

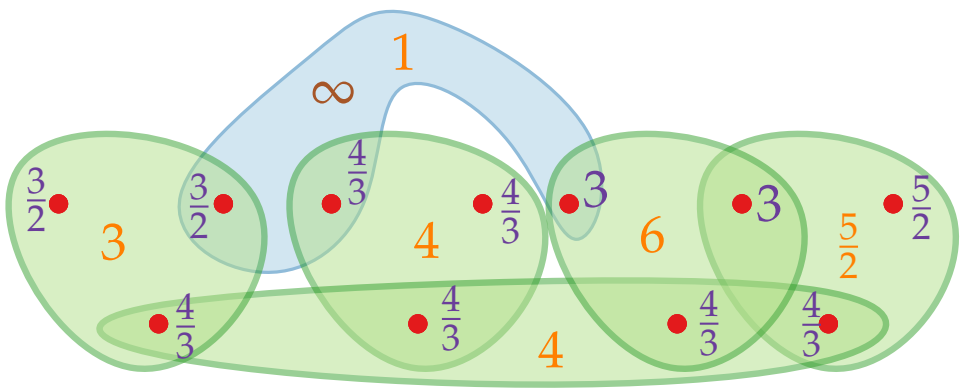
**Observation.** For each $u \in U$, $\text{price}(u)$ is a dual variable
But this dual solution is in general not feasible.

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

*Dual-Fitting-Trick:*

Scale duale variables such that no set is overpacked.

Take $y_u = \text{price}(u)/H_n$.




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

# 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.

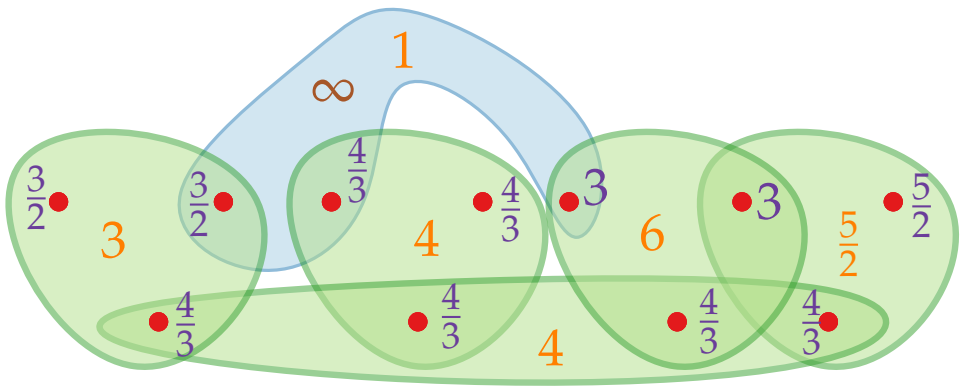
HA: Construct instance where some $S$ are "overpacked" by factor $\approx H_{|S|}$.

*Dual-Fitting-Trick:*

Scale duale variables such that no set is overpacked.

Take $y_u = \text{price}(u)/H_n$.

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




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

# 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.

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

*Dual-Fitting-Trick:*

Scale duale variables such that no set is overpacked.

Take $y_u = \text{price}(u)/H_n$.

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

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

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

**Proof.**

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

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

**Proof.** To prove: No set is overpacked by $y$.

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

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

**Proof.** To prove: No set is overpacked by $y$.
Let $S \in \mathcal{S}$ and $k = |S|$.

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

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

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

**Proof.** To prove: No set is overpacked by $y$.
Let $S \in \mathcal{S}$ and $k = |S|$.

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

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

**Proof.** To prove: No set is overpacked by $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$\Rightarrow y_{u_i} \leq$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1}$$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1} \Rightarrow \sum_{i=1}^{k} y_{u_i} \leq$$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1} \Rightarrow \sum_{i=1}^{k} y_{u_i} \leq \frac{c(S)}{H_n} \cdot \Big( \qquad \Big)$$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1} \Rightarrow \sum_{i=1}^{k} y_{u_i} \leq \frac{c(S)}{H_n} \cdot \left(\frac{1}{k} + \cdots + \frac{1}{1}\right)$$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1} \Rightarrow \sum_{i=1}^{k} y_{u_i} \leq \frac{c(S)}{H_n} \cdot \overbrace{\left(\frac{1}{k} + \cdots + \frac{1}{1}\right)}$$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1} \Rightarrow \sum_{i=1}^{k} y_{u_i} \leq \frac{c(S)}{H_n} \cdot \overbrace{\left(\frac{1}{k} + \cdots + \frac{1}{1}\right)}^{H_k}$$

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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 $y$.

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

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

Consider the iteration in which $u_i$ is covered

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

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

$$\Rightarrow y_{u_i} \leq \frac{c(S)}{H_n} \cdot \frac{1}{k-i+1} \Rightarrow \sum_{i=1}^{k} y_{u_i} \leq \frac{c(S)}{H_n} \cdot \overbrace{\left(\frac{1}{k} + \cdots + \frac{1}{1}\right)}^{H_k} \leq c(S)$$

□

**Lemma.**
The vector $y = (y_u)_{u \in U}$ 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-$H_n$-approximation algorithm for SetCover, where $n = |U|$.

# Result for Dual Fitting

**Theorem.** GreedySetCover is a factor-$H_n$-approximation algorithm for SetCover, where $n = |U|$.

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

# Result for Dual Fitting

**Theorem.** GreedySetCover is a factor-$H_n$-approximation algorithm for SetCover, where $n = |U|$.

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

# Result for Dual Fitting

**Theorem.** GreedySetCover is a factor-$H_n$-approximation algorithm for SetCover, where $n = |U|$.

**Proof.** $\text{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \text{price}(u) = H_n \cdot \sum_{u \in U} y_u \leq$

# Result for Dual Fitting

**Theorem.** GreedySetCover is a factor-$H_n$-approximation algorithm for SetCover, where $n = |U|$.

**Proof.**

$$\text{ALG} = c(\mathcal{S}') \leq \sum_{u \in U} \text{price}(u) = H_n \cdot \sum_{u \in U} y_u \leq$$

$$\leq H_n \cdot \text{OPT}_{\text{relax}}$$

# Result for Dual Fitting

**Theorem.** GreedySetCover is a factor-$H_n$-approximation algorithm for SetCover, where $n = |U|$.

**Proof.**

$$\begin{aligned} \text{ALG} = c(\mathcal{S}') &\leq \sum_{u \in U} \text{price}(u) = H_n \cdot \sum_{u \in U} y_u \leq \\ &\leq H_n \cdot \text{OPT}_{\text{relax}} \\ &\leq H_n \cdot \text{OPT} \end{aligned}$$

□