# Wrap-Up
# CSCI 532

# Test 3 Logistics

1. During class on Thursday 4/30.
2. You can bring your book and any notes you would like, but no electronic devices.
3. You may assume anything proven in class or on homework.
4. Three questions (10 points):
    1) Easy Approximation Algorithm (5 points).
    2) Conceptual Question (3 points).
    3) Hard Approximation Algorithm (2 points).
    4) Bonus (1 point).

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Algorithm: Assign each task to the worker with the smallest current work time.

# Work Scheduling

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

$A_w$ - Tasks assigned to worker $w$.

$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Goal: Show $\frac{1}{w} \sum_i t_i \leq OPT$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.

$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $=$ ?

Goal: Show $\frac{1}{w}\sum_i t_i \leq OPT$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.

$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i$

Goal: Show $\frac{1}{w} \sum_i t_i \leq OPT$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.

$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time

Goal: Show $\frac{1}{w} \sum_i t_i \leq OPT$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.

$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

Goal: Show $\frac{1}{w} \sum_i t_i \leq OPT$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Goal: Show $\frac{1}{w}\sum_i t_i \leq OPT$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it?

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

**This holds when task $j$ is assigned and at the end of the schedule, since $ALG - t_j$ does not change as more tasks are scheduled, but $T_w$'s may get larger.**

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:
$\sum_w (ALG - t_j) \leq \sum_w T_w$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

$$\sum_w \left(ALG - t_j\right) \leq \sum_w T_w$$

$$\Rightarrow \left(ALG - t_j\right) \leq \frac{1}{w}\sum_w T_w$$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.

$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

$$\sum_w (ALG - t_j) \leq \sum_w T_w$$

$$\Rightarrow (ALG - t_j) \leq \frac{1}{w}\sum_w T_w$$

**Does this relate to this?**

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

$$\sum_w (ALG - t_j) \leq \sum_w T_w$$

$$\Rightarrow (ALG - t_j) \leq \frac{1}{w}\sum_w T_w = \frac{1}{w}\sum_i t_i$$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

$$\sum_w (ALG - t_j) \leq \sum_w T_w$$

$$\Rightarrow (ALG - t_j) \leq \frac{1}{w}\sum_w T_w = \frac{1}{w}\sum_i t_i \leq OPT$$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

$$\sum_w (ALG - t_j) \leq \sum_w T_w$$

$$\Rightarrow (ALG - t_j) \leq \frac{1}{w}\sum_w T_w = \frac{1}{w}\sum_i t_i \leq OPT$$

$$\Rightarrow ALG \leq OPT + t_j$$

# Work Scheduling

$A_w$ - Tasks assigned to worker $w$.
$T_w = \sum_{i \in A_w} t_i$ - Worker $w$'s work time.

Given a set of $n$ tasks and $w$ identical workers, assign all tasks such that the longest work time is minimized.

Total task completion time $= \sum_i t_i =$ Total work time $\leq w\ OPT$

$$\Rightarrow \sum_i t_i \leq w\ OPT \Rightarrow \frac{1}{w}\sum_i t_i \leq OPT$$

Consider the worker that dictates $ALG$ ($W_{ALG}$). Why was $W_{ALG}$'s last task $j$ assigned to it? $ALG - t_j \leq T_w$, for all workers.

Sum up all work done by all of the workers:

$$\sum_w (ALG - t_j) \leq \sum_w T_w$$

$$\Rightarrow (ALG - t_j) \leq \frac{1}{w}\sum_w T_w = \frac{1}{w}\sum_i t_i \leq OPT$$

$$\Rightarrow ALG \leq OPT + t_j \leq 2\ OPT$$

**Someone in $OPT$ had to process $t_j$, so $OPT \geq t_j$.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \ldots, v_n$ and weights $w_1, \ldots, w_n$, select the most valuable combination with total weight $\leq W$.

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \ldots, v_n$ and weights $w_1, \ldots, w_n$, select the most valuable combination with total weight $\leq W$.

What if we could take fractional items?

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \dots, v_n$ and weights $w_1, \dots, w_n$, select the most valuable combination with total weight $\leq W$.

What if we could take fractional items?

$$\Rightarrow OPT \leq OPT_{frac}$$

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

**Because $OPT_{frac}$ is allowed to take full items too.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \dots, v_n$ and weights $w_1, \dots, w_n$, select the most valuable combination with total weight $\leq W$.

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

What if we could take fractional items?

$OPT_{frac} = OPT_{greedy}$ up until the final item.

$$\Rightarrow OPT \leq OPT_{frac}$$

**Nothing can be larger than including full items of highest ratio.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \dots, v_n$ and weights $w_1, \dots, w_n$, select the most valuable combination with total weight $\leq W$.

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

What if we could take fractional items?

$OPT_{frac} = OPT_{greedy}$ up until the final item.

$$\Rightarrow OPT \leq OPT_{frac} \leq OPT_{greedy} + v_j$$

**Including the full last item provides more value than including part of it.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \dots, v_n$ and weights $w_1, \dots, w_n$, select the most valuable combination with total weight $\leq W$.

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

What if we could take fractional items?

$OPT_{frac} = OPT_{greedy}$ up until the final item.

$$\Rightarrow OPT \leq OPT_{frac} \leq OPT_{greedy} + v_j$$
$$\leq OPT_{greedy} + v_{max}$$

**Highest valued item is even larger.**

# Knapsack

Knapsack: Given a set of $n$ items with values $v_1, \dots, v_n$ and weights $w_1, \dots, w_n$, select the most valuable combination with total weight $\leq W$.

**Algorithm: Select highest value-weight items until no space is left OR select the single highest valued item.**

What if we could take fractional items?

$OPT_{frac} = OPT_{greedy}$ up until the final item.

$$\Rightarrow OPT \leq OPT_{frac} \leq OPT_{greedy} + v_j$$
$$\leq OPT_{greedy} + v_{max}$$
$$\Rightarrow \max\{OPT_{greedy}, v_{max}\} \geq \frac{1}{2} OPT$$

**They can't both be smaller than half of the value they sum to.**

# Randomized Rounding

# Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Example:

$$U = \{1, 4, 7, 8, 10\}$$

$$S = \{\{1, 7, 8\}, \{1, 4, 7\}, \{7, 8\}, \{4, 8, 10\}\}$$

$\{\{1, 7, 8\}, \{4, 8, 10\}\}$ ✓

$\{\{1, 4, 7\}, \{7, 8\}\}$ ✗

# Set Cover ILP

Objective: $\min \sum_s x_s$

Subject to: $\sum_{s:\, u \in s} x_s \geq 1$, for each $u \in U$

$0 \leq x_s \leq 1$, for each set $s$

+

1 - Add set $s$ to our subset $S_{ALG}$ with probability of $x_s$.

2 - Repeat step 1 $T$-times, while adding sets to $S_{ALG}$.

# Set Cover ILP

Objective: $\min \sum_s x_s$
Subject to: $\sum_{s:\, u \in s} x_s \geq 1$, for each $u \in U$
$0 \leq x_s \leq 1$, for each set $s$

+

1 - Add set $s$ to our subset $S_{ALG}$ with probability of $x_s$.

2 - Repeat step 1 $T$-times, while adding sets to $S_{ALG}$.

Suppose $T = \ln 4n$, where $|U| = n$.

Markov's Inequality:
$\Pr[X \geq a] \leq \frac{E[X]}{a}$

1. What is the size of the solution?

$$E[ALG] = E\left[\sum_{s \in S_{ALG}} X_s\right]$$
$$\leq E\left[\sum_{t \leq T} \sum_{s \in S_{ALG_t}} X_s\right], \text{ since multiple iterations may select } s$$
$$= T \sum_{s \in S} x_s^*, \text{ by previous bound } E\left[\sum_{s \in S_{ALG}} X_s\right] = \sum_{s \in S} x_s^*$$

Thus, $\Pr[ALG > 4 \ln 4n \, OPT] \leq \Pr[ALG > 4T \sum_{s \in S} x_s^*]$

$$\leq \frac{E[ALG]}{4T \sum_{s \in S} x_s^*} = \frac{T \sum_{s \in S} x_s^*}{4T \sum_{s \in S} x_s^*} = \frac{1}{4}$$

# Set Cover ILP

Objective: $\min \sum_s x_s$
Subject to: $\sum_{s:\, u \in s} x_s \geq 1$, for each $u \in U$
$0 \leq x_s \leq 1$, for each set $s$

+

1 - Add set $s$ to our subset $S_{ALG}$ with probability of $x_s$.

2 - Repeat step 1 $T$-times, while adding sets to $S_{ALG}$.

Suppose $T = \ln 4n$, where $|U| = n$.

2. What is the probability solution is valid?

Let $S_u$ be sets of $S$ that contain element $u$.

$$\Pr[u \text{ not covered by } S_{ALG}] = \prod_{t \leq T} \Pr[u \text{ not covered by } S_{ALG_t}]$$
$$\leq \prod_{t \leq T} \frac{1}{e}, \text{ by previous bound}$$
$$= \frac{1}{e^T} = \frac{1}{e^{\ln 4n}} = \frac{1}{4n}$$

---

Thus, $\Pr[S_{ALG} \text{is not a cover}] = \Pr[\bigcup_u (u \text{ not covered by } S_{ALG})]$
$$\leq \sum_u \Pr[u \text{ not covered by } S_{ALG}], \text{ by union bound}$$
$$\leq \sum_u \frac{1}{4n} = n \frac{1}{4n} = \frac{1}{4}$$

# Set Cover ILP

Solution is "good" (valid and $ALG \leq O(\ln(n))OPT$) with probability $\geq \frac{9}{16} \approx .56$

Good enough? What if we run the algorithm two independent times?

Probability some run is good $= 1 - \left(\frac{7}{16}\right)^2 \approx 0.81$

Three times? $1 - \left(\frac{7}{16}\right)^3 \approx 0.92$

Four times? $1 - \left(\frac{7}{16}\right)^4 \approx 0.96$

Ten times? $1 - \left(\frac{7}{16}\right)^{10} \approx 0.9997$

**High likelihood of "good" solution with guaranteed polynomial time algorithm.**

I.e., Exponential improvement $\left(1 - \left(\frac{7}{16}\right)^t\right)$ for polynomial time work $(t)$.

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation

**for** $t = 0$ to $\ln 4n$

**add** $s$ to $S_{ALG}$ with probability $x_s^*$

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation

**for** $t = 0$ to $\ln 4n$

**add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover
    **for** $t = 0$ to $\ln 4n$
        **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover
  **for** $t = 0$ to $\ln 4n$
    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^*$

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover
  **for** $t = 0$ to $\ln 4n$
    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^*$

**Guarantee of "good" solution with high likelihood polynomial time?**

$$\sum_{s \in S} x_s^* = OPT_{LP} \leq OPT$$

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover
  **for** $t = 0$ to $\ln 4n$
    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

**Guarantee of "good" solution with high likelihood polynomial time?**

$$\sum_{s \in S} x_s^* = OPT_{LP} \leq OPT$$

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover
  **for** $t = 0$ to $\ln 4n$
    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

**This is all stuff we know!**

**Guarantee of "good" solution with high likelihood polynomial time?**

$$\sum_{s \in S} x_s^* = OPT_{LP} \leq OPT$$

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation

**while** $S_{ALG}$ is not a cover **or** $ALG > 4 \ln 4n \sum_{s \in S} x_s^*$

  **for** $t = 0$ to $\ln 4n$

    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.

Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

**This is all stuff we know!**

**Guarantee of "good" solution with high likelihood polynomial time?**

$$\sum_{s \in S} x_s^* = OPT_{LP} \leq OPT$$

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover **or** $ALG > 4 \ln 4n \sum_{s \in S} x_s^*$
    **for** $t = 0$ to $\ln 4n$
        **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

After first iteration of the while loop:

$$\Pr[S_{ALG} \text{ is not a cover}] \leq \frac{1}{4}$$

$$\Pr[ALG > 4 \ln 4n \, OPT] \leq \frac{1}{4}$$

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover **or** $ALG > 4 \ln 4n \sum_{s \in S} x_s^*$
    **for** $t = 0$ to $\ln 4n$
        **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

After first iteration of the while loop:

$$\Pr[S_{ALG} \text{ is not a cover}] \leq \frac{1}{4}$$

$$\Pr[ALG > 4 \ln 4n \, OPT] \leq \frac{1}{4}$$

$\Rightarrow$Probability while loops to second iteration

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation

**while** $S_{ALG}$ is not a cover **or** $ALG > 4 \ln 4n \sum_{s \in S} x_s^*$

  **for** $t = 0$ to $\ln 4n$

    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.

Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

After first iteration of the while loop:

$$\Pr[S_{ALG} \text{ is not a cover}] \leq \frac{1}{4}$$

$$\Pr[ALG > 4 \ln 4n \, OPT] \leq \frac{1}{4}$$

$\Rightarrow$Probability while loops to second iteration

$= \Pr[S_{ALG} \text{ is not a cover}]$ OR $\Pr[ALG > 4 \ln 4n \, OPT]$

**Guarantee of "good" solution with high likelihood polynomial time?**

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation

**while** $S_{ALG}$ is not a cover **or** $ALG > 4 \ln 4n \sum_{s \in S} x_s^*$

  **for** $t = 0$ to $\ln 4n$

    **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.

Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

After first iteration of the while loop:

$$\Pr[S_{ALG} \text{ is not a cover}] \leq \frac{1}{4}$$

$$\Pr[ALG > 4 \ln 4n \, OPT] \leq \frac{1}{4}$$

**Guarantee of "good" solution with high likelihood polynomial time?**

$\Rightarrow$Probability while loops to second iteration

$$= \Pr[S_{ALG} \text{ is not a cover}] \text{ OR } \Pr[ALG > 4 \ln 4n \, OPT]$$

$$\leq \Pr[S_{ALG} \text{ is not a cover}] + \Pr[ALG > 4 \ln 4n \, OPT] = \frac{1}{2}$$

# Set Cover ILP

$x_s^*$ = optimal solutions to LP relaxation
**while** $S_{ALG}$ is not a cover **or** $ALG > 4 \ln 4n \sum_{s \in S} x_s^*$
    **for** $t = 0$ to $\ln 4n$
        **add** $s$ to $S_{ALG}$ with probability $x_s^*$

We can easily test if $S_{ALG}$ is actually a valid cover or not.
Suppose that $ALG \leq 4 \ln 4n \sum_{s \in S} x_s^* \leq 4 \ln 4n \, OPT$.

After first iteration of the while loop:

$$\Pr[S_{ALG} \text{ is not a cover}] \leq \frac{1}{4}$$

$$\Pr[ALG > 4 \ln 4n \, OPT] \leq \frac{1}{4}$$

**Guarantee of "good" solution with high likelihood polynomial time?**

$\Rightarrow$ Probability while loops to second iteration

$$= \Pr[S_{ALG} \text{ is not a cover}] \text{ OR } \Pr[ALG > 4 \ln 4n \, OPT]$$

$$\leq \Pr[S_{ALG} \text{ is not a cover}] + \Pr[ALG > 4 \ln 4n \, OPT] = \frac{1}{2}$$

$\Rightarrow E[\#\text{ while loop iterations}] = 2$ (geometric distribution)

# Set Cover ILP

**High likelihood of "good" solution with guaranteed polynomial time algorithm.**

Monte Carlo Algorithm: Randomness effects correctness, not runtime.

**Guarantee of "good" solution with high likelihood polynomial time.**

Las Vegas Algorithm: Randomness effects runtime, not correctness.