1 What is a Linear Programming Problem?

A *linear program* (LP) is a minimization problem where we are asked to minimize a given linear function subject to one or more linear inequality constraints. The linear function is also called the *objective function*.

Formulation:

$$Minimize \sum_{i=1}^{n} C_i X_i \qquad \text{(where } C_i \in \Re \text{ and are constants and } X_i \in \Re \text{ and are variables)}$$

Subject to constraints:

$$a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \geq b_1$$
$$a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \geq b_2$$
$$a_{31}x_1 + a_{32}x_2 + ... + a_{3n}x_n \geq b_3$$
$$\vdots$$
$$a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n \geq b_n$$

Alternately, we can rewrite the above formulation as:

$$\text{Minimize } C^T X \qquad \text{(where } C, X \in \Re \text{ and are column vectors)}$$

Subject to constraints:

$$AX \geq b \qquad \text{(where } b \in \Re^m, A \in \Re^{m \times n}\text{)}$$

Given C, A and b the above LP can be solved in time $poly(inputlength)$.

2 Set Cover using LP rounding

For the definition of the Set Cover problem and examples, refer to Valika's scribed notes. In the greedy set cover problem we were given (X, F) earlier. Now we consider (X, F, c) where c is the cost function for every set F.
$c : F \longrightarrow \Re^+$
$F \subseteq 2^X$
$|X| = n$
Thus $\forall S \in F, c(S) =$"cost" of using set S in the cover.
Earlier, for the greedy set cover explained by Valika, we can consider $\forall S, c(S) = 1$.

2.1 LP formulation

Find a cover X of minimum cost where $C \subseteq F$ is a cover if $\bigcup_{\forall S \in C} S = X$

Let us assign a binary variable x_S for each $S \in F$ where

$$x_S = \begin{cases} 1 & S \in C \\ 0 & S \notin C \end{cases}$$

Subject to constraints:

$$\sum_{S \in F, S \ni \alpha} x_S \geq 1 \qquad \text{for each } \alpha \in X \tag{1}$$

$$x_S \in \{0, 1\} \qquad \text{for each } S \in F \tag{2}$$

Note: Above problem is not an LP since above statement is not a linear constraint. The above problem is actually an Integer Linear Problem or IP. Solving an IP is NP-Complete.

Let us now relax our second constraint to $0 \leq x_S \leq 1$ for each i and allow $x_i \in \Re$.

Here we are violating our original inequality direction, since $x_i \leq 1$
We can easily fix this problem by restating our constraint (2) as:

$$x_S \geq 0 \qquad (\text{for each } S \in F)$$

$$-x_S \geq -1 \qquad (\text{for each } S \in F)$$

Now we can use the cost function to differentiate our set cover returned from the earlier greedy version which had no concept of the cost of a set:

$$Minimize \sum_{S \in F, S \ni \alpha} c(S) x_S \geq 1 \qquad \text{for each } \alpha \in X$$

Proposed Algorithm: LP-Set-Cover

Algorithm 1: LP-SET-COVER(V, E)

1 Construct LP relaxation for given instance (X, F, c) as shown above
2 Invoke polynomial time LP solver to get a solution
$P = (P_{S_1}, P_{S_2}, ...P_{S_t}), where\{S_1, S_2, ..., S_t\} = F$
3 $C \longleftarrow \emptyset$
4 **for** each $S \in F$ **do**
5 $\quad C \longleftarrow C \cup \{S\}$ with probability P_S
6 Repeat steps 3 & 4 $(\ln n + 10)$ times and return unions of the covers computed in $\hat{C}$
7 **return** $\hat{C}$

2.2 Analysis of Algorithm

Let us now verify that the above algorithm is correct and analyze its optimality.
Expected Cost of the Solution:

$$\begin{aligned}
E[cost(\hat{C}] &= E[\sum_{S \in F} c(S) x_S] \qquad \text{where } x_S \text{ iff } S \in \hat{C} \\
&= \sum_{S \in F} c(S) E[x_S] \\
&= \sum_{S \in F} c(S) Pr[S \in \hat{C}] \\
&\leq \sum_{S \in F} c(S)(\ln n + 10) P_S \\
&= (\ln n + 10) \sum_{S \in F} c(S) P_S \\
&= (\ln n + 10) OPT_{LP} \\
&\leq (\ln n + 10) OPT_{IP}
\end{aligned}$$

Therefore LP-Set-Cover is a $(\ln n + 10)$ Approx Algorithm (In the expected sense)

2.3 Correctness:

What is the probability that $\hat{C}$ is a cover i.e what is the Pr[$\hat{C}$ is a cover] ?
Pick any $\alpha \in X$. Let S_α be a set of sets such that $\{S \in F : S \ni \alpha\} = \{T_1, T_2, ..T_k\}$
Let us find the probability that α was not picked in the set picked in steps 3-4 of LP-Set-Cover

$$\begin{aligned}
Pr[\alpha \text{not covered}] &= (1 - P_{T_1})(1 - P_{T_2})......(1 - P_{T_k}) \\
&\leq \left(\frac{(1 - P_{T_1}) + (1 - P_{T_2}) + ... + (1 - P_{T_k})}{k} \right)^k \qquad \text{By AM-GM inequality} \\
&= \left(1 - \frac{P_{T_1} + P_{T_2} + ... + P_{T_k}}{k} \right)^k \\
&\leq \left(1 - \frac{1}{k} \right)^k \\
&\leq e^{-1}
\end{aligned}$$

Now we find Probability after $(\ln n + 10)$ repetitions of steps 3-4 to get our cover $\hat{C}$

$$\begin{aligned}
Pr[\alpha \text{ not covered by } \hat{C}] &\leq (e^{-1})^{(\ln n+10)} \\
&= \frac{e^{-10}}{n} \\
Pr[\exists \alpha \in X \text{ not covered by } \hat{C}] &\leq n.\frac{e^{-10}}{n} \\
&= e^{-10} \\
\rightarrow Pr[\hat{C} \text{ is a cover}] &\geq 1 - e^{-10} \approx 0.999
\end{aligned}$$

2.4 Final Algorithm:

Repeat LP-Set-Cover untill we have a cover. Expected number of repetitions < 2
Note: Repeating does not affect quality of approximation, just the running time.