CS 466/666: Algorithm Design and Analysis **University of Waterloo: Fall 2025**

Lecture 10

October 7, 2025

Instructor: Sepehr Assadi

Topics of this Lecture

In this lecture, we will examine approximation algorithms and integrality gaps.

1 Integrality Gaps, Rounding, and Approximation Algorithms

Let us now go back again to the applications of LPs for designing algorithms for combinatorial optimization problems. So far, we examined bipartite matching/vertex cover and in both case it turned out that the optimal fractional solution can be *rounded* to an integral solution (a maximum matching or a minimum vertex cover, respectively) without changing the value at all.

Nevertheless, in many scenarios, we cannot expect to have such a *lossless* rounding. For instance, consider the general (non-bipartite) vertex cover problem wherein the input graph can be arbitrary. You might already have a strong suspicion that we should not be able to round the value of LP without any loss to an integral vertex cover. After all, we can solve LPs in polynomial time while (general) vertex cover is an NP-hard problem; so, if optimal value of vertex cover LP is always equal to the minimum vertex cover, then we get a polynomial time algorithm for an NP-hard problem. While, technically speaking, this is not known to be impossible[1], it is still something to be extremely suspicious of. We will examine this in more details in the rest of this lecture.

1.1 Minimum Vertex Cover and Its LP

Consider the following LP for the minimum (non-bipartite) vertex cover problem on a graph $G = (V, E)$:

$$\begin{aligned} \min_{y \in \mathbb{R}^V} \quad & \sum_{v \in V} y_v \\ \text{subject to} \quad & y_u + y_v \geqslant 1 \qquad \forall\, (u, v) \in E, \end{aligned}$$

[1]Of course, we do not know whether P = NP or not yet ...

$$y_v \geqslant 0 \qquad\qquad \forall\, v \in V.$$

We define the **integrality gap** of this LP as the largest ratio between the minimum vertex cover and the optimal value of the LP (roughly speaking, the maximum value of how "costly" this relaxation is). Formally, the integrality gap of the vertex cover LP is:

$$\sup_G \quad \frac{\text{size of minimum vertex cover of } G}{\text{optimal value of vertex cover LP on } G}.$$

So our question from before is what is the integrality gap of the minimum vertex cover problem? We claim it is almost 2, i.e., there are graphs where size of the integral minimum vertex cover is (almost) twice as large as the optimum solution of the LP. A very simple example is a complete graph on n vertices:

- The solution $y_v = 1/2$ for all $v \in V$ is a feasible fractional to the LP: for every edge $(u, v) \in E$, we have $y_u + y_v = 1/2 + 1/2 = 1$ and thus the edge is covered. Hence, the optimal LP value is $\leqslant n/2$ (we did not prove the optimality of this solution, only its feasibility, hence the '$\leqslant$' sign and not '$=$' although the latter is also true; try proving it yourself).
- On the other hand, the size of minimum vertex cover is $n - 1$; if we do not pick any two vertices u and v in the vertex cover, then the edge between them is not covered.

This implies that the integrality gap on n-vertex graph is

$$\frac{n-1}{n/2} = 2 - \frac{1}{n}.$$

Given the supremum definition of integrality gap, for $n \to \infty$, we have that the integrality gap is at least 2.

But what about an upper bound? That is the topic of **approximation algorithms**, i.e., finding a way of rounding the fractional LP so that we do not lose "too much" in the value (in general, approximation algorithms try to find a solution which is "approximately" optimal – in the formal sense, this means that we can bound the ratio between the value of the returned solution and the optimal solution to the problem).

Let us show that integrality gap of vertex cover LP is at most 2 also (which combined with the previous part implies it is precisely 2).

Proposition 1. *Given any feasible solution $y \in \mathbb{R}^V$ to the vertex cover LP, there is a polynomial time algorithm that returns a vertex cover of size at most $2\sum_{v \in V}$.*

Proof. Define the set $U \subseteq V$ as:

$$U := \{v \in V \mid y_v \geqslant 1/2\},$$

i.e., the set of vertices with y-value at least $1/2$.

Firstly, we argue that U is a valid vertex cover of G. This is because for any edge $e = (u, v) \in E$, we have $y_u + y_v \geqslant 1$ (by the feasibility of LP) which means at least one of u or v has y-value at least $1/2$ and is thus added to the set U. So, every edge has at least one endpoint in U and thus U is a vertex cover.

Secondly, we can bound the size of U as follows:

$$\begin{aligned}
|U| &= \sum_{v \in V} \mathbb{I}(v \in U) && \text{(where } \mathbb{I}(v \in U) = 1 \text{ if } v \in U \text{ and 0 otherwise)} \\
&= \sum_{v \in V} \mathbb{I}(y_v \geqslant 1/2) && \text{(where } \mathbb{I}(\cdot) \text{ in a similar way as above)} \\
&\leqslant \sum_{v \in V} 2 \cdot y_v,
\end{aligned}$$

$$\text{(because either } y_v < 1/2 \text{ and thus } \mathbb{I}(y_v \geqslant 1/2) = 0 \text{ or } y_v \geqslant 1/2 \text{ and thus } \mathbb{I}(v \geqslant 1/2) = 1 \leqslant 2y_v)$$

concluding the proof. □

Proposition 1 now implies that integrality gap of the vertex cover LP is at most 2 because we can round any optimal solution to get a vertex cover of size at most twice as large. But this also tells us something very useful algorithmically: we have a polynomial time algorithm for 2-approximation of minimum vertex cover. Firstly, solve the LP optimally in polynomial time to get the optimal solution y^* and then run Proposition 1 to turn it into a vertex cover of size at most $2\sum_v y^*_v$. But recall that optimal value of LP is always a lower bound on the minimum vertex cover (as we relaxed the integer linear program to an LP and thus can only improve the optimal solution). Thus, this algorithm also outputs a vertex cover of size at most twice that of minimum vertex cover.

Remark. There are much easier ways of getting a 2-approximation algorithm to minimum vertex cover. We will see those later in the course. That being said, the above approach has its positives, for instance, it is quite robust and versatile; you can extend it as it is to the *weighted* version of the problem where the vertices have weights and we want to pick the cover with minimum weight. Just apply the same exact strategy to the following weighted vertex cover LP instead:

$$\begin{aligned} \min_{y \in \mathbb{R}^V} \quad & \sum_{v \in V} w(v) \cdot y_v \\ \text{subject to} \quad & y_u + y_v \geqslant 1 \qquad \forall\, (u,v) \in E, \\ & y_v \geqslant 0 \qquad \forall\, v \in V, \end{aligned}$$

where $w(v)$ in the objective is the given weight of the vertex $v \in V$.

2 The Set Cover Problem

We now see another famous example of approximation algorithms, this time for the **set cover** problem. In the set cover problem, we have a collection of m sets $S_1, \ldots, S_m$ from a universe U of n elements. We further assume that each set S has a weight $w(S) > 0$. The goal is to find the minimum weight collection of sets that *cover* the universe U, i.e.,

$$\min_{T \subseteq [m]} \sum_{i \in T} w(S_i) \quad \text{such that} \quad \bigcup_{i \in T} S_i = U.$$

This is a very famous NP-hard problem with tons of applications. It also generalizes various problems, including the minimum vertex cover, we saw in the previous part.

The following is an LP relaxation of the set cover problem.

$$\begin{aligned} \min_{x \in \mathbb{R}^m} \quad & \sum_{S} w(S) \cdot x_S \\ \text{subject to} \quad & \sum_{S \ni e} x_S \geqslant 1 \qquad \forall\, e \in U, \\ & x_S \geqslant 0 \qquad \forall\, S. \end{aligned}$$

To see why this is a relaxation, consider the case when $x_S \in \{0,1\}$ for all S with the interpretation that $X_S = 1$ iff we pick S in the solution; then, the objective is minimizing the weight of the solution and each constraint ensures that we pick at least one set that covers this particular element in the solution.

In the previous section, we considered a *deterministic* rounding scheme for the minimum vertex cover problem. We can do a similar type of argument for the set cover LP also. However, the bounds we get are going to be quite weak in general and depend on the maximum number of sets that cover any fixed element (basically the maximum "length" of the summation in the constraints of the above LP). You are encouraged to work out the details of this rounding scheme on your own.

3 Randomized Rounding for Set Cover

We now consider another simple rounding scheme for the set cover LP, this time by randomly rounding each set to 1 depending on the value the LP assigned to it.

In particular, suppose we solve the set cover LP and obtain a *fractional* solution $x \in \mathbb{R}^m$. Then, it is natural to think that the sets S with "large" values of x_S are "more important" than the ones with lower values of x_S. So, if we want to pick a set to include in our solution, we may want to prioritize picking sets with larger values of x_S over the smaller ones. However, instead of doing this deterministically, we are going to do it randomly, namely, sample each set S with a probability p_S proportional to x_S, i.e.,

$$p_S \propto x_S.$$

The question is then what to pick for the scaling factor? Let us pick a parameter $\beta \geqslant 1$ for this, i.e., sample each set S with probability $p_S := \beta \cdot x_S$ (where x_S is the solution obtained from the LP). I.e.

Algorithm 1. A randomized rounding algorithm for set cover.

1. Solve the LP relaxation of the set cover problem to obtain the optimal fractional solution $x \in \mathbb{R}^m$.
2. For every set S, add S to the final solution, denoted by ALG, with probability

$$p_S := \min(\beta \cdot x_S, 1)$$

where β is a parameter to be determined later[a].

[a] If you really cannot wait for later, β is going to be $2 \ln n$ ...

The analysis of this algorithm is in two steps. We first show that we can always upper bound the expected approximation ratio of the algorithm for any choice of β. In the following, we use $w(ALG)$ to denote the weight of the sets in ALG, i.e., the weight of the solution returned by the algorithm.

Claim 2. *$\mathbb{E}[w(ALG)] = \beta \cdot opt_{LP}$, where opt_{LP} is the optimum objective value of the LP.*

Proof. We have,

$$\begin{aligned}
\mathbb{E}[w(ALG)] &= \mathbb{E}\left[\sum_S \mathbb{I}(S \in ALG) \cdot w(S)\right] && \text{(because we pay a weight of } w(S) \text{ for every set } S \in ALG) \\
&= \sum_S w(S)\, \mathbb{E}[\mathbb{I}(S \in ALG)] && \text{(by linearity of expectation)} \\
&= \sum_S w(S) \Pr(S \in ALG) \\
& \qquad \text{(as the expected value of an indicator random variable is its probability of being one)} \\
&= \sum_S w(S) \cdot p_S && \text{(by the definition of } p_S) \\
&= \sum_S w(S) \cdot \beta \cdot x_S && \text{(by the choice of } \beta) \\
&= \beta \cdot opt_{LP} && \text{(by the definition of the objective value of the LP)}
\end{aligned}$$

concluding the proof. □

Claim 2 ensures that we can always bound the expected weight of our solution in terms of the optimal LP for any choice of β that we decide to pick. We now establish a lower bound on the probability that ALG output by the algorithm is an actual set cover, i.e., is a feasible solution.

Claim 3. $\Pr\left(ALG \textit{ is a } \underline{\textit{not}} \textit{ a feasible set cover}\right) \leqslant n \cdot e^{-\beta}$.

Proof. By union bound,

$$\begin{aligned}\Pr\left(ALG \text{ is a } \underline{\text{not}} \text{ a feasible set cover}\right) &= \Pr\left(\text{there exists an element not covered by } ALG\right) \\ &\leqslant \sum_{e \in U} \Pr\left(ALG \text{ does not cover } e\right).\end{aligned}$$

Fix any $e \in U$ and let us bound the probability term for e in the RHS above. Note that we can assume without loss of generality that for every S that covers e, $p_S < 1$ as if $p_S = 1$, we will deterministically cover e and so this probability is zero. We have,

$$\begin{aligned}\Pr\left(ALG \text{ does not cover } e\right) &= \prod_{S \ni e} (1 - p_S) \\ &\qquad \text{(none of the sets } S \ni e \text{ should be picked, and the choices are independent)} \\ &\leqslant \prod_{S \ni e} \exp\left(-p_S\right) && \text{(as } 1 - x \leqslant e^{-x} \text{ for all } x \in (0,1)) \\ &= \exp\left(-\sum_{S \ni e} p_S\right) && \text{(as } e^a \cdot e^b = e^{a+b} \text{ for all } a, b) \\ &= \exp\left(-\sum_{S \ni e} \beta \cdot x_S\right) && \text{(as we have } p_S = \beta \cdot x_S \text{ by our assumption that } p_S < 1) \\ &\leqslant \exp\left(-\beta \cdot 1\right) && \text{(as } x \text{ is a feasible solution to the LP and thus } \textstyle\sum_{S \ni e} x_S \geqslant 1) \\ &= e^{-\beta}.\end{aligned}$$

The bound in the claim now follows as there are n elements in total. □

Claim 3 now implies that if we pick $\beta = 2 \ln n$, then, with probability $1 - 1/n$, we obtain a feasible set cover as well (and that if $\beta = o(\ln n)$, we are really not going to get a feasible cover – prove this for yourself).

Finally, we can also apply Markov bound to Claim 2 and say that for $\beta = 2 \ln n$,

$$\Pr\left(w(ALG) > 8 \ln n \cdot opt_{LP}\right) \leqslant 1/4.$$

So, by union bound, with probability $1 - (1/4 + 1/n) \geqslant 2/3$, we obtain a feasible set cover which is an $O(\log n)$ approximation. This concludes our randomized rounding algorithm for set cover.

4 Dual Fitting for Set Cover

We now consider another, less direct, approach for using linear programming to obtain an approximation algorithm for set cover. Instead of first solving the LP and then rounding it, we are going to design a natural greedy algorithm for set cover that on the surface has nothing to do with LPs. But, to analyze this algorithm, we will rely on the notion of *LP duality* that we introduced in the previous lecture.

To continue, we first state, without proof, the following LP as the dual of the set cover LP (you are encouraged to prove this is indeed the dual):

$$\begin{aligned} \max_{y \in \mathbb{R}^n} \quad & \sum_{e \in U} y_e \\ \text{subject to} \quad & \sum_{e \in S} y_e \leqslant w(S) && \forall\, S, \\ & y_e \geqslant 0 && \forall\, e. \end{aligned}$$

Let us now consider the following combinatorial algorithm.

Algorithm 2. A greedy algorithm for set cover

```
1. Let $L = U$ originally be the elements *left* to cover and $ALG = \emptyset$ be the final set cover.
2. While $L \neq \emptyset$
    (a) Let $S$ be any set that minimizes
            $\frac{w(S)}{|L \cap S|}$,
        i.e., the ratio of weight of $S$ to the number of new elements it covers.
    (b) Add $S$ to $ALG$ and update $L \leftarrow L \setminus S$.
3. Return $ALG$ as a set cover.
```

The fact that ALG is indeed a feasible set cover is immediate to see. Thus, we only need to bound the approximation value of this algorithm. To do, we are going to use the dual LP. We first define an *infeasible* dual "solution" y' that we can easily relate the cost of our algorithm to, and then show that with a proper scaling down, we can turn y' into a feasible dual solution y. The final bound then is obtained by applying weak duality theorem.

We define $y' \in \mathbb{R}^n$ as follows. In the algorithm, whenever an element e is covered *for the first time* by a set S, i.e., the iteration that we remove e from L, we define

$$y'_e := \frac{w(S)}{|L \cap S|},$$

where L is the set of elements before we pick S in ALG.

Claim 4. $w(ALG) = \sum_{e \in U} y'_e$.

Proof. We have,

$$\sum_{e \in U} y'_e = \sum_{S \in ALG} \sum_{\substack{e \text{ covered} \\ \text{first by } S}} y'_e \qquad \text{(every element is covered exactly once for the 'first time' in } ALG\text{)}$$

$$= \sum_{S \in ALG} \sum_{\substack{e \text{ covered} \\ \text{first by } S}} \frac{w(S)}{|L \cap S|}$$

(by definition of y'_e where L here refers to the set of remaining elements before picking S in ALG)

$$= \sum_{S \in ALG} |S \cap L| \cdot \frac{w(S)}{|L \cap S|}$$

(because the number of elements covered by S for the first time is $|L \cap S|$)

$$= \sum_{S \in ALG} w(S) = w(ALG),$$

concluding the proof. □

So, we can relate the cost of our algorithm to the infeasible dual solutions y'. We now show that a scaled down version of y' actually leads to a feasible dual solution.

Claim 5. *Define* $y = \frac{y'}{\ln(n)+1}$. *Then,* y *is a feasible dual solution.*

Proof. To prove y is a feasible solution, we need to show that for every set S,

$$\sum_{e \in S} y_e \leqslant w(S).$$

Let $S = \{e_1, e_2, \ldots, e_k\}$ where we additionally assume that e_1 is covered before (or at the same time as) e_2, e_2 before (or at the same time as) e_3, and so on and so forth. When e_1 was covered, the "cost" of set S that we minimize in the algorithm greedily was $w(S)/k$ as all its elements are still in L. Since we are picking a set with minimum "cost" and by the definition of y', we have

$$y'_{e_1} \leqslant \frac{w(S)}{k}.$$

Then, when we pick e_2, we have,

$$y'_{e_2} \leqslant \frac{w(S)}{k-1},$$

because again S can cover at least $k-1$ elements and thus the "cost" of the set we pick is at most the RHS above. Continuing like this, for every $i \in [k]$, we have,

$$y'_{e_i} \leqslant \frac{w(S)}{k-i+1}.$$

Thus,

$$\sum_{i=1}^{k} y'_{e_i} \leqslant \sum_{i=1}^{k} \frac{w(S)}{k-i+1} = w(S) \cdot \sum_{i=1}^{k} \frac{1}{i} \leqslant w(S) \cdot (\ln k + 1) \leqslant w(S) \cdot (\ln n + 1),$$

where the first inequality is by the sum of Harmonic series and the second is because $k \leqslant n$. By the definition of $y = y'/(\ln(n) + 1)$, we obtain that $\sum_e y_e \leqslant w(S)$ as desired, concluding the proof. □

To conclude, we have,

$$\begin{aligned} w(ALG) &= \sum_e y'_e && \text{(by Claim 4)} \\ &= (\ln(n) + 1) \cdot \sum_e y_e && \text{(by the definition of } y\text{)} \\ &\leqslant (\ln(n) + 1) \cdot opt_{dual} \\ & && \text{(because } y \text{ is a feasible solution to dual LP by Claim 5 which is a maximization LP)} \\ &\leqslant (\ln(n) + 1) \cdot opt_{LP}. && \text{(by weak duality theorem)} \end{aligned}$$

This concludes the proof as opt_{LP} is at most that of the (integral) set cover.

Thus, we obtained a deterministic $(\ln(n) + 1)$ approximation algorithm to weighted set cover this way.

Remark. In general, a dual fitting approach to a minimization LP works by starting with a basic (combinatorial) algorithm for the problem and using LPs only in its analysis. Using the dual LP of the problem, we show that the primal (integral) solution we picked is "paid for" by some *infeasible* dual. By fully paid for we mean that the objective function value of the primal solution found is at most the objective function value of the dual computed. The main step in the analysis consists of scaling the dual by a suitable factor and showing that the shrunk/expanded dual is now indeed feasible. This dual solution then provides a bound on the optimal solution, and the factor in this bound is the approximation guarantee of the algorithm.