CS 224: Advanced Algorithms Spring 2017

Lecture 12 — March 2, 2017

Prof. Jelani Nelson *Scribe: Kevin Zhang*

1 Overview

Last lecture, we covered approximate complementary slackness, the basics of primal/dual linear programs, and gave an initial analysis of set cover using primal/dual LPs. Today, we'll wrap up our approximate solution to set cover, introduce approximation algorithms using LPs, discuss integrality gaps between LPs and IPs, and introduce a few examples of PTAS, FPTAS, and FPRAS.

2 Set Cover Wrap-Up

The set cover problem consists of a set of n items $\mathcal{O}$ and a set of m subsets $\mathcal{C} \subseteq \mathcal{O}^*$. We want to choose some sets $\mathcal{K} \subseteq \mathcal{C}$ such that $\bigcup_{k \in \mathcal{K}} k = \mathcal{O}$ and $|\mathcal{K}|$ is minimal. The primal (left) and dual (right) are given below:

$$\begin{aligned} &\min \sum_{S \in \mathcal{C}} x_S \\ &\forall e \in [n], \sum_{S:\ e \in S} x_S \geq 1 \\ &x \geq 0 \end{aligned} \qquad \begin{aligned} &\max \sum_{e=1}^{n} y_e \\ &\forall S \in \mathcal{C}, \sum_{e \in S} y_e \leq 1 \\ &y \geq 0 \end{aligned}$$

Last time, we showed an $\ln m$**-competitive fractional algorithm** that solves online set cover. We need to fix this now so that our solution satisfies integrality. The below turns out to be true:

Theorem 2.1. *It is possible to obtain a solution with a* $\ln m \cdot \ln n$*-competitive ratio for online set cover.*

Proof. We obtain such a ratio by taking an approach similar to the solution we used last lecture to the ski rental problem. With the ski rental problem, we maintained a variable x that decided whether we bought skis or not. To handle integrality, we picked a random α uniformly between 0 and 1; then as soon as $x > \alpha$, we bought skis. The expected cost of doing so maintained the same cost from the fractional solution, so we maintained the same competitive ratio. In our set cover solution, we implement a similar approach:

- maintain x, y as in the solution from last lecture,
- pick $\alpha_S \in [0,1]$ uniformly for each $S \in \mathcal{C}$,
- and take S into the solution when $x_S > \alpha_S$.

Claim 2.1.1. *The expected # of sets taken in the output from the algorithm above is at most* $\ln m \cdot OPT$.

This is easy enough to prove; we have:

$$\mathbb{E}(\#\text{ sets taken}) = \sum_S \mathbb{P}(S \text{ taken}) = \sum_S \mathbb{P}(\alpha_S < x_S) = \sum_S x_S \leq \ln m \cdot (\sum_e y_e) = \ln m \cdot \text{OPT}$$

However, this solution fails to account for the fact that the solution might not be feasible! Each x_S is not guaranteed to be one (it has exactly probability x_S of being 1). It turns out to be fairly straightforward to come up with examples where, with high probability, one of the items will not be covered. We fix this by choosing multiple alphas for each set instead:

Claim 2.1.2. *For any $p \in [0,1]$, there exists an algorithm which fails to give a feasible primal solution with probability at most p.*

We modify our algorithm above to accomplish this. Let $q = \ln n + \ln 1/p$. We choose $\alpha_S(k)$ uniformly for $1 \leq k \leq q$ for each S, and take the set S into our solution if $x_S > \alpha_s(k)$ for any $1 \leq k \leq q$. By union bound, the probability of any particular S being taken in becomes at most qx_S, which gives an expected number of taken sets of $\ln m \cdot q \cdot \text{OPT}$. By taking q α's for each set, the probability that the solution is not feasible becomes:

$$\begin{aligned}
\mathbb{P}(\text{not feasible}) &= \mathbb{P}(\exists e \in [n] \text{ s.t. no taken sets contain } e) \\
&= \sum_e \prod_{S:\ e \in S} \mathbb{P}(\text{didn't take } S) \\
&= \sum_e \prod_{S:\ e \in S} (1 - x_S)^q \\
&\leq n \prod_{S:\ e \in S} (1 - x_S)^q \\
&\leq n \prod_{S:\ e \in S} e^{-q x_S} \\
&= n \exp(-q \sum_{S:\ e \in S} x_S) \\
&\leq e^{-q} \cdot n = \boxed{p}
\end{aligned}$$

where we use $1 + z \leq e^z$ for all $z \in \mathbb{R}$ and $\sum_{S:\ e \in S} x_S \geq 1$ (a constraint from the primal LP).

This solution gives a competitive ratio of $\ln m \cdot (\ln n + \ln 1/p) = \boxed{\ln m \cdot \ln n}$ for some fixed p.

□

The above turns out to be optimal, up to a $\log\log n + \log\log m$ factor. Note that an alternative (suggested in class by a student) is to continually try larger and larger q's while we fail to find a feasible solution in order to guarantee feasibility; the expected competitive ratio will still be on the order of $\ln m \cdot \ln n$.

3 Approximation Algorithms

We now cover a class of problems where we don't really care about finding an exact solution. This may be because the problem is too hard to solve (e.g. NP-hard). We generally formulate such problems in terms of minimizing/maximizing some objective. Our approach is then to (e.g. in the case of minimization), find a solution with value at most $(1+\epsilon)\cdot \text{OPT}$ quickly.

Such an example takes the form of weighted set cover. In this form of set cover, choosing a set S has cost c_S instead, and we want to minimize the total cost while covering all elements. In the dual, we have $y_e \le c_S$ instead of $y_e \le 1$. It's known that set cover with costs 1 is already NP-hard, so we can't hope to solve weighted set cover more quickly. The new primal (left) and dual (right) are below:

$$\min \sum_{S\in\mathcal{C}} c_S x_S \qquad\qquad \max \sum_{e=1}^{n} y_e$$
$$\forall e \in [n], \sum_{S:\ e\in S} x_S \ge 1 \qquad\qquad \forall S \in \mathcal{C}, \sum_{e\in S} y_e \le c_S$$
$$x \ge 0 \qquad\qquad y \ge 0$$

It turns out that the canonical greedy algorithm is an $\ln n$-competitive approximation for uniform set cover in polynomial time (proven by Johnson [1] and Lovász [2]). A later proof by Chvátal [3] proved using primal/dual analysis that a greedy approach can give a $\ln n$ approximation even with general costs. Feige [4] showed that a solution to SAT in $O(n^{O(\log\log n)})$ time can be reduced to a $\approx 0.99 \ln n$ approximation for weighted set cover that runs in polynomial time.

We now show the $\ln n$-approximation solution to weighted set cover.

3.1 Weighted Set Cover

This algorithm doesn't need to maintain the dual solution in order to come up with primal feasible solution; we only maintain the dual solution for our analysis later. Alternative algorithms look at the dual solution to make decisions with the primal, but we will not cover those here. Our algorithm works as follows:

- initialize x, y to zero
- while there exists an uncovered element $e \in [n]$,

– choose a set S with minimum ratio c_S/k_S, where k_S is the number of newly covered elements
– for every newly covered element e, set $y_e = c_S/k_S$

Theorem 3.1. *The above algorithm gives a* $\ln n$*-competitive solution.*

Proof. Notice that the primal solution is integral; we don't have to worry about fractional values for the primal! Furthermore, the algorithm maintains equal primal and dual values; every time we run through the while loop, the primal cost increases by c_S for some S, while the dual profit increases by c_S/k for k elements, giving an overall increase of c_S as well. So we've found a feasible solution for the primal - what about the dual?

Claim 3.1.1. *The above algorithm gives a feasible primal solution, and a feasible dual solution by dividing* y *by* H_n*.*

The primal solution is clearly feasible. To show that the dual solution is feasible, we need to prove that $\forall S$, we have $\sum_{e \in S} y_e \leq H_n c_S$. Let $|S| = k$ for some S. Order the elements in S by the time in which they were first covered, so that we have $S = \{e_1, e_2, \cdots, e_k\}$.

Claim 3.1.2. *For a particular set* S *and a particular* $e_i \in S$ *(where* i *is its index after the elements of* S *are ordered by the time they are covered),* $y_{e_i} \leq \frac{c_S}{k-i+1}$*.*

This is fairly straightforward to show; consider the exact iteration in which e_i is covered. Before this iteration, S has not been taken. Taking S would set $y_{e_i} = \frac{c_S}{k-i+1}$; but since, at each iteration, we take the set with the minimum ratio of cost to # of newly covered elements, we must have $y_{e_i} \leq \frac{c_S}{k-i+1}$.

Now note that, since $y_{e_i} \leq \frac{c_S}{k-i+1}$, we have:

$$\begin{aligned}
\sum_{i=1}^{k} y_{e_i} &\leq \sum_{i=1}^{k} \frac{c_S}{k-i+1} \\
&= \sum_{j=1}^{k} \frac{c_S}{j} \\
&= c_S H_k \\
&\leq c_S H_n
\end{aligned}$$

as desired. Note that $H_n = \ln n + \mathrm{O}(1)$, so we have a $\ln n$-competitive solution, as desired. Furthermore, this analysis shows that we can actually obtain a $\ln K$-competitive solution, where $K = \max_{S \in \mathcal{C}} |S|$!

□

The above proof made use of a technique called **dual fitting**. With dual fitting, we build primal and dual solutions simultaneously to find a primal feasible solution, then scale the dual by some factor to make it feasible. This gives a fairly straightforward way to compute a tentative competitive ratio!

We now give another example of an approximate algorithm using primal/dual LPs.

3.2 Unweighted Vertex Cover

In unweighted vertex cover, we have an undirected graph $G = (V, E)$ of vertices and edges. We want to find some $V' \subseteq V$ such that every edge $e = (u, v) \in E$ has at least one of u, v in V', and $|V'|$ is minimal. The primal (left) and dual (right) are given below:

$$\begin{aligned} \min \sum_{v \in V} x_v \\ \forall e = (u, v) \in E, x_u + x_v \geq 1 \\ x \geq 0 \end{aligned} \qquad \begin{aligned} \max \sum_{e \in E} y_e \\ \forall v \in V, \sum_{e:\ e=(v,)|(,v)} y_e \leq 1 \\ y \geq 0 \end{aligned}$$

Note that this is a special case of set cover; the elements here are edges, and every vertex corresponds to a set that consists of all edges connected to that vertex. Consider the following (very simple!) algorithm

- initialize $V' = \emptyset$
- while there are edges remaining uncovered, pick one edge $e = (u, v)$ and insert both u, v into V'

Theorem 3.2. *The above algorithm is a 2-approximation for unweighted vertex cover.*

Proof. For every edge $e = (u, v) \in E$, the optimal solution V^* either contained u or v. But our solution V' either contains one or both, so $|V'| \leq 2|V^*|$, and we are done. □

It might seem like we could do better than a 2-approximation; however, the following discussion will show this is optimal.

4 LP Integrality Gaps

Up to this point, we've been building integral solutions x such that $c^T x \leq \alpha \cdot \text{OPT}$. However, for some LPs, there is inherently a gap between the best you can do with an integral solution and a fractional solution. Let Q be some instance of a particular LP. In this case, using our approach thus far, we cannot find an approximation ratio for this LP better than

$$\max_Q \frac{\text{OPT}_{\text{integral}}}{\text{OPT}_{\text{fractional}}}$$

We give some concrete examples of gaps we cannot overcome below.

4.1 Unweighted Vertex Cover

Consider vertex cover with a triangle graph. In this case, the optimal integer solution must take exactly 2 vertices, giving an optimum of 2. However, a fractional solution can simply assign $x_v = 1/2$ for all vertices, giving an optimum of 3/2 and an integrality gap of at least 4/3. This means that we can't get better than a 4/3-approximation!

In particular, if we look at G_k, the complete graph on k vertices, integral solutions must take at least $k-1$ vertices; however, fractional solutions need only assign 1/2 to all vertices, giving an integrality gap of $\frac{2(k-1)}{k} \to 2$ for large k. Thus, the optimal approximation ratio for this problem is 2.

Note: the validity of this proof relies on something called the Unique Games Conjecture; it turns out that this conjecture proves the optimality of many types of approximation algorithms. The Wikipedia article has a good description of the conjecture (along with material pertaining to the vertex cover problem!).

4.2 Set Cover

Note that a gap of 2 for vertex cover implies that set cover has a gap of at least 2 as well. However, the integrality gap for set cover could be even worse (and it is!). In fact, we have the following:

Theorem 4.1. *The previous LP formulation of set cover has an integrality gap of* $\Omega(\log n)$.

Proof. We give a specific instance of set cover that requires a gap of $\log_2 n$. Define F_2^k as the set of all k-dimensional vectors with component values drawn from $\mathbb{Z}_2 = \{0, 1\}$. Consider a specific q, and let the set of elements be $E = F_2^q / \{0\}$ (so that E consists of all bit vectors of length q except zero). Notice then that $n = |E| = 2^q - 1$. Then consider a set $\mathcal{C}$ of sets, each of which is based on an element in E so that $\mathcal{C}$ consists of all sets $S_\alpha = \{e \in E : \langle e, \alpha \rangle \equiv 1 \mod 2\}$, where $\alpha \in F_2^q$. The notation $\langle e, \alpha \rangle$ simply means the dot product of e and α. We first prove some properties of S_α.

Claim 4.1.1. *For any particular $z \in E$ and $\alpha \in F_2^q$, $\mathbb{P}(\langle z, \alpha \rangle \equiv 1 \mod 2) = 1/2$.*

This should be straightforward to see; to be rigorous, take any index i where $z_i = 1$ (this must be possible since $z \in E$ and therefore $z \neq 0$) and let z_{-i} be the vector z with the ith component removed. Suppose that for any $\alpha \in F_2^q$, we have $\mathbb{P}(\langle z_{-i}, \alpha_{-i} \rangle \equiv 1 \mod 2) = p$ for some p, and therefore $\mathbb{P}(\langle z_{-i}, \alpha_{-i} \rangle \equiv 0 \mod 2) = 1 - p$.

Then, since every component of α is distributed uniformly randomly over $\{0, 1\}$ when α is picked uniformly randomly from F_2^q, $\mathbb{P}(z_i \alpha_i \equiv 1) = \mathbb{P}(z_i \alpha_i \equiv 0) = 1/2$. Then the probability that $\langle z, \alpha \rangle \equiv 1$ is simply the probability that $z_i \alpha_i$ is 1 and the dot product of the remaining components is zero, plus the probability that $z_i \alpha_i$ is 0 and the dot product of the remaining components is one, or $\frac{1}{2}(1 - p) + \frac{1}{2}p = 1/2$, as desired.

We use this to develop a fractional solution with cost 2 as follows:

Claim 4.1.2. *Setting $x_S = \frac{2}{|\mathcal{C}|}$ gives a feasible fractional solution with cost 2.*

Claim 4.1.1 means that any $z \in E$ is in exactly half the sets in $\mathcal{C}$! Then:

$$\sum_{S \in \mathcal{C}} x_S = |\mathcal{C}| \cdot \frac{2}{|\mathcal{C}|} = 2$$

and for any $e \in E$, we have

$$\sum_{S:\ e \in S} x_S = \sum_{S:\ e \in S} \frac{2}{|\mathcal{C}|} = \frac{|\mathcal{C}|}{2} \cdot \frac{2}{|\mathcal{C}|} = 1 \geq 1$$

meaning this solution is feasible. It remains to show that the optimal integral solution to this instance requires a cost of at least $\log n$.

Claim 4.1.3. *The optimal integral solution to the previous LP formulation of set cover requires at least q sets.*

Suppose we can cover all elements with exactly t sets, $\{S_{e_1}, S_{e_2}, \cdots, S_{e_t}\}$; we bound t later. Since we were able to cover all elements with these sets, this implies that the intersection of their complements contains exactly the zero vector, or:

$$\bigcap_{i \in [t]} S_{e_i}^C = \{0\}$$

But if there are no elements not contained in these sets, this implies that every $z \in E$ satisfies t

linear equations:

$$\begin{aligned}\langle z, e_1\rangle &= 0\\ \langle z, e_2\rangle &= 0\\ &\cdots\\ \langle z, e_t\rangle &= 0\end{aligned}$$

But since z is drawn from a q-dimensional space, and each of these equations reduces the solution space by at most 1 dimension, the solution space must have dimension at least $q - t$. Since the solution space is assumed to be zero (there are no uncovered elements), we must have $t \geq q$, and the claim is proven.

Finally, since the optimal integral solution requires at least q sets, and the optimal fractional solution requires cost 2, the integrality gap is at least $\Omega(q/2) = \Omega(\log_2 n/2) = \boxed{\Omega(\log n)}$, as desired. □

Note that this implies that our discussion previously already gave an optimal approximation to set cover (up to a constant). Furthermore, this discussion relied on a specific formulation of set cover; it is very possible that an alternative formulation of the set cover problem will have a different integrality gap! This discussion only proves that, given our current formulation, we can do no better than a $\log n$ approximation ratio to the true optimum. It turns out that there is a system that is occasionally more useful that linear programming, called semi-definite programming (which we will see next lecture!).

Finally, we end with a discussion of non-LP approximation algorithms.

5 PTAS, FPTAS, FPRAS Approximations

These terms, respectively, mean:

- Polynomial time approximation scheme: for all ϵ, we can find a solution to a minimization problem with value $(1 + \epsilon) \cdot \text{OPT}$ in time at most $f(\epsilon)n^{g(\epsilon)}$.
- Fully-polynomial time approximation scheme: like PTAS, except we can get time $\text{poly}(n, 1/\epsilon)$.
- Fully-polynomial time randomized approximation scheme: like FPTAS, except we only require that we succeed with probability at least $2/3$.

Below, we give a PTAS solution to the knapsack problem.

5.1 Knapsack

In the knapsack problem, we are given a knapsack with weight W, and some set of n objects each with some weight and value. We want to find some set of objects with maximum total value subject to the constraint that their total weight is at most W. Furthermore, all weights and values are

integral. The LP is given below:

$$\max \sum_{i \in [n]} x_i v_i$$
$$\sum_{i \in [n]} x_i w_i \leq W$$
$$x, w, v \geq 0$$

We can get a $O(nW)$ runtime simply by defining the subproblem $f(i, b)$ as the max value obtainable with the first i objects and remaining weight capacity b; recursion on $f(i, b)$ gives the desired result. Similarly, we can get time $O(nV)$ by defining the subproblem $g(i, b)$ as the minimum weight of any items whose values sum to b (sort of like a dual!) and recursing on $g(i, b)$. However, the above algorithms are only **pseudo-polynomial** because the inputs to this problem can be expressed using $\log W + \log \sum_{i=1}^{n} w_i + v_i$ space (and these runtimes are therefore exponential in the input size).

The natural greedy approach would be to sort items by "how good they are" (in terms of the ratio between value and weight), and continually pick items in decreasing order. However, this approach trips up whenever the item weights and values are not small relative to W; in particular, letting $n = 2$, $v_1 = 2, w_1 = 1, v_2 = W, w_2 = W$ shows that this approach doesn't work (we'd pick object 1 instead of 2, when clearly picking object 2 is better). However, this algorithm lends itself to an alternative approach:

Theorem 5.1. *A PTAS exists for the knapsack problem with poly$(n) \cdot n^{1/\epsilon}$ runtime and approximation ratio 2.*

This PTAS always acts by either being greedy or taking the most valuable item (whichever is optimal). We'll prove this works, next time.

References

[1] David S. Johnson. Approximation Algorithms for Combinatorial Problems. *J. Comput. Syst. Sci.*, 9(3): 256–278 (1974).

[2] László Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13(4): 383–390 (1975).

[3] Vašek Chvátal. A Greedy Heuristic for the Set-Covering Problem. *Mathematics of Operations Research*, 4(3): 233–235 (1979).

[4] Uriel Feige. A Threshold of $\ln n$ for Approximating Set Cover. *Journal of the ACM*, 45(4): 634–652 (1998).