Advanced Approximation Algorithms **(CMU 18-854B, Spring 2008)**

Lecture 2: LP Relaxations, Randomized Rounding

Jan 17, 2008

Lecturer: Ryan O'Donnell *Scribe: Ali Kemal Sinop*

1 Introduction

In the last lecture, a greedy $\lceil \log n \rceil$-factor approximation algorithm was given for Set-Cover, which was also extended to give a 1 vs $1 - \frac{1}{e}$-approximate search algorithm for Max-Coverage problem.

Theorem 1.1. *[1] $\forall \epsilon > 0$, approximating Set-Cover within a factor of $(1-\epsilon)\ln n$ is hard. Also 1 vs $1 - \frac{1}{e} + \epsilon$ decision problem for Max-Coverage is NP-hard, too.*

In this lecture, we will try to answer two questions.

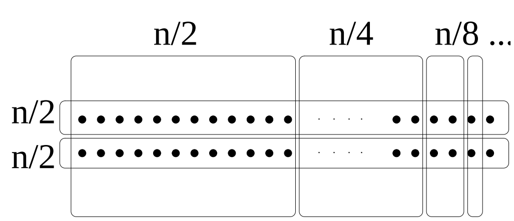
- Does greedy algorithm do better?
- Can we do better?

Before answering the first question for Set-Cover problem, we will give a formal definition for gap instances.

Definition 1.2 (c vs s algorithmic gap instance)**.** An instance where the optimum value is at most c and the given algorithm achieves a value of at least s (for minimization problems).

2 Set Cover

For set cover problem, when the optimal cover size c is 1, the greedy algorithm will return this set. However for $c = 2$, consider the following example:



Assume that greedy algorithm breaks ties in the worst way. It can be seen that the optimal solution is 2. However, greedy algorithm will take $\log_2 n$ sets. Therefore the gap for this instance is $\frac{\ln n}{2 \ln 2}$.

To get this bound sharper, we can consider a similar instance for $c = k$, where there are k sets covering all elements and one set covers $\frac{1}{k}$-fraction of all elements, next one covers $\frac{1}{k}$-fraction of the remaining elements and so on. Then for these instances, the gap is k vs $\log_{\frac{1}{1-\frac{1}{k}}} n$. Since

$$\frac{1}{1-\frac{1}{k}} = 1 + \frac{1}{k} + \Theta(\frac{1}{k^2})$$

$$\ln(1+\epsilon) = \epsilon + \Theta(\epsilon^2)$$

$$\ln(\frac{1}{1-\frac{1}{k}}) = \frac{1}{k} + \Theta(\frac{1}{k^2})$$

Hence the ratio is $\frac{\ln n}{k \ln \frac{1}{1-\frac{1}{k}}} \approx \frac{\ln n}{k \frac{1}{k}} = (1 - \Theta(\frac{1}{k})) \ln n$, which answers the first question in a negative way. In the next section, another approximation algorithm for Min Set-Cover problem will be introduced.

2.1 Linear Programming Relaxation

First we will introduce the weighted version of Min-SetCover problem.

Weighted Min Set-Cover: Given base elements $E = \{e_1, \ldots, e_n\}$, sets $S_1, \ldots, S_m \subseteq E$, and cost for each set, $c(S_i) \geq 0$, find a cover for base elements whose total cost is minimum.

Consider the following Integer-Programming (IP) formulation for this problem. There is an indicator variable, x_S associated with every set S, which takes value 1 if set S is selected, 0 otherwise.

$$\begin{aligned} \text{minimize} \quad & \sum_S x_S c(S) && \text{(objective)} \\ & x_S \geq 0, \quad \forall S && \text{(constraints)} \\ & \sum_{S \ni e} x_S \geq 1, \quad \forall e \in E \\ & x_S \in \mathbb{Z} && \text{(integrality constraint)} \end{aligned}$$

Notice that there is no need to introduce an explicit constaint for $x_S \leq 1$. It is obvious that every solution of this problem is a feasible set cover, and every set cover is a feasible solution for this problem. Therefore the optimal value of this problem is equal to Min-SetCover, which means solving this IP is NP-hard.

General methodology for solving these IP programs is to relax them to a linear program (LP). We will replace the integrality constraint $x_S \in \mathbb{Z}$ with $x_S \in \mathbb{R}^+$. Due to [2], we know that LP $\in$ P.

Definition 2.1 ($\mathsf{Lp}(\mathcal{I})$)**.** Given an NP-hard optimization problem instance $\mathcal{I}$, and an LP relaxation, we write $\mathsf{Lp}(\mathcal{I})$ for the optimal value of LP problem.

Key Observation: Since all feasible solutions for IP are also feasible solutions for LP relaxation, we have $\mathsf{Lp}(\mathcal{I}) \leq \mathsf{Opt}(\mathcal{I})$ for minimization problems and $\mathsf{Lp}(\mathcal{I}) \geq \mathsf{Opt}(\mathcal{I})$ for maximization problems.

For set cover problem, if we solve LP relaxation, we would get an optimal solution $x_S^* \in [0,1]$ such that $\forall e$, $\sum_{S \ni e} x_S^* \geq 1$. So we need a way of converting this fractional solution to an integral one, which is called rounding.

2.2 Randomized Rounding [3]

The general algorithm for randomized rounding is:

1. For each set S, include it with probability x_S^*.
2. Repeat (1) t times, and take the union (however in an actual implementation, one needs to repeat (1) only until all sets are covered).

For every execution of (1), the expected cost is:

$$\mathbf{E}[\text{cost of (1)}] = \mathbf{E}[\sum_S c(S) 1_{\{\text{S is chosen}\}}] = \sum_S c(S) \mathbf{Pr}[\text{S is chosen}] = \sum_S c(S) x_S^* = \mathsf{Lp}(\mathcal{I})$$

Total cost is $\mathbf{E}[\text{overall cost}] \leq t \cdot \mathsf{Lp} \leq t \cdot \mathsf{Opt}$ (due to the linearity of expectation). Notice that $\mathbf{E}[\text{overall cost}]$ might be strictly less than $t \cdot \mathsf{Lp}$, as we might pick the same set multiple times but we only need to pay for it once. Since cost is non-negative, using Markov's inequality, we have $\mathbf{Pr}[\text{total cost} \geq (1 + \frac{2}{\ln n}) t \cdot \mathsf{Opt}] \leq \frac{1}{1 + \frac{2}{\ln n}} \leq 1 - \frac{2}{\ln n} \leq 1 - \frac{1}{\ln n}$.

If we let $t = \ln n + C \ln \ln n$, $(1 + \frac{2}{\ln n})(\ln n + C \ln \ln n) = \ln n + O(\ln \ln n)$, we have

$$\mathbf{Pr}[\text{total cost} \leq (\ln n + O(\ln \ln n)) \cdot \mathsf{Opt}] \geq \frac{1}{\ln n} \tag{1}$$

For any element e,

$$\mathbf{Pr}[\text{e is uncovered after (1)}] = \prod_{S \ni e} (1 - x_S^*) < \prod_{S \ni e} \exp(-x_S^*) = \exp(-\sum_{S \ni e} x_S^*) \leq \exp(-1).$$

Hence $\mathbf{Pr}[\text{e is uncovered after (2)}] \leq \exp(-t) \leq \frac{1}{n \ln^C n}$. Since there are n elements,

$$\mathbf{Pr}[\text{any element is uncovered}] \leq n \frac{1}{n \ln^C n} = \frac{1}{\ln^C n} \tag{2}$$

Combining Eq.s 1 and 2, we get (for $C = 2$),

$$\mathbf{Pr}[\text{all elements covered} \wedge \text{total cost} \leq (\ln n + O(\ln \ln n)) \cdot \mathsf{Opt}] \geq \frac{1}{\ln n} - \frac{1}{\ln^2 n} = \Omega(\frac{1}{\ln n})$$

Thus we can use the result from Problem 1a of Homework 1 to obtain an algorithm which would run in polynomial time and output a correct solution with probability at least $1 - 2^{-n}$.

In what ways is this algorithm better or worse than greedy algorithm?

Worse: Slower, slightly worse approximation factor.

Better: Works for weighted case (though greedy algorithm also does with a small modification), $\ln n$ guarantee against the value of $\mathsf{Lp} \leq \mathsf{Opt}$, where *potentially* $\mathsf{Lp} \ll \mathsf{Opt}$.

Recall that an algorithm is called α-factor approximation algorithm if it is guaranteed to have a value $\leq \alpha \cdot \mathsf{Opt}$.

Definition 2.2 (α-factor LP-approximation algorithm)**.** Algorithm is guaranteed to output a solution $\leq \alpha \cdot \mathsf{Lp}$.

Definition 2.3 (c vs. s LP-approximation algorithm)**.** Whenever $\mathsf{Lp} \leq c$, the algorithm gives a solution $\leq s$.

Idea: Almost all approximation algorithm guarantees are actually (LP,SDP,...)-approximation guarantees. Or in other words, are performance guarantees not against Opt but some poly-time computable lower bound function on Opt. Why? It seems like math (and humans) can only reason about polynomial time stuff.

Definition 2.4 (c vs. s integrality gap instance)**.** For an instance $\mathsf{Lp} \leq c$ but $\mathsf{Opt} \geq s$.

For Min-SetCover problem, $\exists (1-\epsilon)\ln n$-factor integrality gap instances.

Notice that, $\mathsf{Opt} \leq$ Above algorithm $\leq (\ln n + o(\ln n))\mathsf{Lp}$, so Opt and Lp differ by a factor of $\ln n$. So we have, for instances with large integrality gap,

$$\overbrace{\mathsf{Alg} \qquad \underbrace{\mathsf{Opt} \qquad\qquad \mathsf{Lp}}_{(1-\epsilon)\ln n}}^{\ln n + o(\ln n)}$$

So it is weird that algorithm thinks it is doing badly on integrality gap instances though it is not. What about algorithm gap instances for LP+randomized rounding? In this case, LP gives the correct optimum value, but algorithm "doesn't know it".

3 Max-Coverage

Now we are going to give a similar LP-relaxation for Max-Coverage problem.

3.1 LP Relaxation

Consider the following formulation, which is similar to the one for Min-SetCover except for variables z_e which denotes whether if element e is covered. Also m is the maximum number of sets to be taken.

$$
\begin{aligned}
\text{maximize} \quad & \tfrac{1}{n}\sum_e z_e & & \text{(objective)} \\
& x_S \geq 0, & \forall S & \quad \text{(constraints)} \\
& \sum_{S \ni e} x_S \geq z_e, & \forall e \in E & \\
& \sum_S x_S \leq m & & \\
& 0 \leq z_e \leq 1 & \forall e &
\end{aligned}
$$

One can do the same thing here till m sets are taken.

3.1.1 Algorithmic gap instance

The hard instances for Max-Coverage are similar for hardness proof. First we will show a $\frac{3}{4}$-gap instance.

Let elements be vertices of a q-hypercube for some large q, ie $e \in \{0,1\}^q$. There are $2q$ sets, such that $S_{i,b} = \{e : e_i = b\}, i \in [q], b \in \{0,1\}$ ($S_{i,b}$ contains all bit strings of length q whose i^{th} bit is b). Let $m = 2$. It is obvious that the solution is $S_{i,0}$ and $S_{i,1}$ for any i.

Consider an optimal solution for this LP problem such that $x^*_{S_{i,b}} = \frac{1}{q}$ and $z^*_e = 1, \forall i \in [q], b \in \{0,1\}, e \in \{0,1\}^q$. Obviously $\mathsf{Lp} = 1$ for this solution.

Randomized rounding will pick any two random sets $S_{i,b}$ and $S_{j,b'}$ (because it can't distinguish between any set). Except with probability $\leq O(\frac{1}{q})$, $i \neq j$. There are 2^{q-2} elements with $e_i \neq b$ and $e_j \neq b'$, so this solution covers at most $\frac{2^q - 2^{q-2}}{2^q} = \frac{3}{4}$ of the elements. By letting ground set to be $\{1, 2, \ldots, k\}^q$, one can generalize this result to $1 - \frac{1}{e}$.

3.2 LP-Integrality Gap Instances for Set-Cover

(from course blog)

As promised, herein I will discuss "LP integrality gap instances" for Set-Cover. Recall that these are instances $\mathcal{I}$ for which $Lp(\mathcal{I})$ is small but $Opt(\mathcal{I})$ is large.

Let's start with one that looks quite similar to the LP+Randomized-Rounding algorithmic gap we saw in class:

Ground elements: all nonzero elements of $\{0,1\}^q$ – thought of as $\mathbb{F}_2^q$, the q-dimensional vector space over the field of size 2. There are $n = 2^q - 1$ ground elements.

Sets: For each $\alpha \in \mathbb{F}_2^q$ we have a set $S_\alpha = \{e \in \mathbb{F}_2^q \setminus \{0\} : \alpha \cdot e = 1\}$. Here $\cdot$ denotes the usual dot product in $\mathbb{F}_2^q$: $\alpha_1 e_1 + ... + \alpha_q e_q$ (mod 2). There are $M = 2^q$ sets. (The set S_0 is kind of pathetic, being the empty set, but never mind.)

Analysis: Each ground element is contained in exactly half of the sets. (Standard undergrad probability exercise.) It follows that in an LP solution, if we take each set fractionally to the extent $2/M$, then each element is fractionally covered to the extent 1.

Hence $Lp(\mathcal{I}) \leq M(2/M) = 2$. (In fact, Lp is less than this...)

On the other hand, we claim $Opt(\mathcal{I}) \geq q$. Suppose by way of contradiction that there are some $q-1$ sets $S_{\alpha_1}, ..., S_{\alpha_{q-1}}$ covering all ground elements. So

$S_{\alpha_1} \cup ... \cup S_{\alpha_{q-1}} = \mathbb{F}_2^q \setminus \{0\}$

$\Leftrightarrow \bigcap_{i=1}^q \overline{S_{\alpha_i}} = \emptyset$

$\Leftrightarrow \bigcap_{i=1}^q \{e \in \mathbb{F}_2^q : \alpha_i \cdot e = 0\} = \{0\}$.

But each set in the intersection above is a hyperplane in $\mathbb{F}_2^q$. This gives the contradiction, because in dimension q, you can't have $q-1$ hyperplanes intersecting on just the point 0.

Conclusion: This instance is a 2 vs. $\log_2{(n+1)}$ integrality gap for Set-Cover. This yields a ratio of $\Omega(\ln n)$, but not the full $(1-\epsilon)\ln n$ we've come to expect.

Comparing with the way things have gone before, I bet you think I'm going to now tell you a straightforward way to extend this to get the full $(1-\epsilon)\ln n$ gap, perhaps by looking at an instance with ground elements $\mathbb{F}_k^n$.

Well, I'm here to tell you that I don't know how to do this. Quite likely it's possible, but I couldn't see how. I'd be very very interested if one of you could give such an example.

So is there another kind of integrality gap instance? Or perhaps Lp and Opt can never be off by more than a $(\log_2 n)/2$ factor...

Since Feige proved that the $(1-\epsilon)\ln n$ decision problem is hard, we know that the former must be the case. (Otherwise, the efficient algorithm of computing and outputting Lp would contradict his theorem.) Here is one way to show it:

Fix an abstract set of ground elements, Ω, of cardinality n. Introduce sets $S_1, ..., S_M$ by picking them randomly (and indendently). For each S_i, we choose it by including each element of Ω with probability $1/k$. We will take $M = (Ck^2)\log n$ for a large C to be named later.

Analysis: Each element is expected to be contained in about a $1/k$ fraction of the sets. If we take C large enough, then a Chernoff bound + union bound will imply that with high probability, every element is contained in at least a $(1/k)(1-\epsilon)$ fraction of sets. (Here we can make ϵ as small as we like by making C large.) Thus we can get a valid fractional solution to the LP by taking each set to the extent $(k/M)/(1-\epsilon)$; this solution has value $k/(1-\epsilon)$. Summarizing: with high probability, $Lp \leq k/(1-\epsilon)$.

We now would like to show that with high probability, $Opt \geq t := (1-\epsilon)k\ln n$. This will give us an Opt/Lp integrality gap of $(1-\epsilon)^2 \ln n$, which is good enough.

To see this, we fix any set of t out of M set indices and show that the probability these sets end up being a valid cover is very small. If the probability is significantly smaller even than $1/\binom{M}{t}$ then we can union bound over all possible size-t covers and be done.

Now each element e has a $1-1/k$ chance of being uncovered by a given set, and thus our fixed t sets have a $(1-1/k)^t$ chance of leaving e uncovered. With our choice of t, this quantity is essentially $\exp(-(1-\epsilon)\ln n) = 1/n^{1-\epsilon}$ (slight cheat here as the inequality goes the wrong way, but one can fix this easily). The events that the t sets cover each element are actually independent across elements. So the probability that all elements are covered is at most $(1-1/n^{1-\epsilon})^n \leq \exp(-n^{\epsilon})$. This is indeed much less than $1/\binom{M}{t}$, since this latter quantity only is of the order $1/(\log n)^{O(\log n)}$ (assuming k and ϵ are "constants"), much smaller than $\exp(-n^{\epsilon})$.

References

[1] U. Feige. A threshold of ln for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[2] L. G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademiia Nauk SSSR*, 224:1093–1096, 1979. English Translation: *Soviet Mathematics Doklady*, Volume 20, pp. 191–194.

[3] P. Raghavan and C. D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.