ORIE 6334 Bridging Continuous and Discrete Optimization Nov. 25, 2019

Lecture 21

Lecturer: David P. Williamson *Scribe: Makis Arsenis*

1 Set Cover Problem

In the (offline) set cover problem, we are given as input n elements denoted by $E = \{e_1, e_2, \ldots, e_n\}$ and m sets $S_1, S_2, \ldots, S_m \subseteq E$ with corresponding weights $w_1, w_2, \ldots w_m \in \mathbb{R}^+$. The goal is to pick a collection of sets indexed by $I \subseteq \{1, 2, \ldots, m\}$ which cover all the elements, i.e. $\bigcup_{j \in I} S_j = E$ while minimizing the total weight $\sum_{j \in I} w_j$.

There is a simple, greedy $O(\log n)$-approximation algorithm for the set cover problem and in fact $O(\log n)$ is the best approximation ratio one can achieve unless $P = NP$. In this lecture we're going to see a randomized approximation algorithm based on an LP relaxation of the problem which can be modified to work for the online version of set cover (see Section 2).

Consider the following LP relaxation of the set cover problem where we introduce a variable x_j for every set S_j. The intended meaning of this variable is that $x_j = 1$ whenever $j \in I$ and $x_j = 0$ otherwise.

$$\text{(P)} \quad \begin{aligned} \min \quad & \sum_{j=1}^{m} w_j x_j \\ \text{s.t.} \quad & \sum_{j: e_i \in S_j} x_j \geq 1, \quad i = 1, \ldots, n \\ & x_j \geq 0, \qquad\qquad j = 1, \ldots, m \end{aligned}$$

The corresponding dual LP is the following:

$$\text{(D)} \quad \begin{aligned} \max \quad & \sum_{i=1}^{n} y_i \\ \text{s.t.} \quad & \sum_{i: e_i \in S_j} y_i \geq w_j, \quad j = 1, \ldots, m \\ & y_i \geq 0, \qquad\qquad i = 1, \ldots, n \end{aligned}$$

Notice that if we denote by OPT the value of the optimal solution to the set cover problem and LP_{OPT} the value of the optimal solution to the primal LP then the following inequalities hold:

$$\text{OPT} \geq \text{LP}_{\text{OPT}} \geq \text{ (any dual feasible solution)}$$

The plan is solve the LP and use the *randomized rounding* technique to obtain an index set I which is a valid set cover with high enough probability while the expected total weight

[0]This lecture is partially based on a survey by Buchbinder and Naor, *Design of Competitive Online Algorithms via a Primal-Dual Approach* [1].

of the chosen sets is most $O(\log n)$ away from the OPT. The following lemmas prove that Algorithm 1 has the above approximation guarantees.

Algorithm 1: Offline Set Cover (randomized) $O(\log n)$-approximation

```
I ← ∅
Solve the LP to get a solution x*
for j ← 1 to m do
    for k ← 1 to 2 ln n do
        Pick X(j, k) ∈ [0, 1] uniformly at random.
    end
    T_j ← min_k X(j, k)
    if x*_j ≥ T_j then
        I ← I ∪ {j}
    end
end
```

Lemma 1

$$\mathbb{E}\left[\sum_{j\in I} w_j\right] \leq (2\ln n)\sum_{j=1}^{m} w_j x_j^* \leq (2\ln n)\,\mathrm{OPT}\,.$$

Proof: Let E_{jk} be the event that $X(j,k) \leq x_j^*$. It's easy to see that $\Pr[E_{jk}] = x_j^*$. Now,

$$\Pr[j \in I] = \Pr\left[\bigcup_k E_{jk}\right] \leq (2\ln n) x_j^*$$

By linearity of expectation:

$$\mathbb{E}\left[\sum_{j\in I} w_j\right] = \sum_{j=1}^{m} \Pr[j \in I] w_j \leq \sum_{j=1}^{m} (2\ln n)\cdot x_j^* w_j = (2\ln n)\mathrm{LP}_{\mathrm{OPT}} \leq (2\ln n)\,\mathrm{OPT}\,.$$

□

Lemma 2

$$\Pr[I \text{ is not a set cover}] = \Pr\left[\bigcup_{j\in I} S_j \neq E\right] \leq \frac{1}{n}$$

Proof: Pick an arbitrary element e_i and compute the probability that it is not covered:

$$\begin{aligned}
\Pr\left[e_i \text{ is not covered}\right] &= \Pr\left[\bigcap_{j:e_i\in S_j}\bigcap_{k=1}^{2\ln n}\overline{E_{jk}}\right] \\
&= \prod_{j:e_i\in S_j}\prod_{k=1}^{2\ln n}(1-x_j^*) \\
&= \prod_{j:e_i\in S_j}(1-x_j^*)^{2\ln n} \\
&\le \left(e^{-\sum_{j:e_i\in S_j}x_j^*}\right)^{2\ln n} \\
&\le e^{-2\ln n} = \frac{1}{n^2},
\end{aligned}$$

where we use the fact that $\sum_{j:e_i\in S_j} x_j^* \ge 1$, since x^* is a feasible LP solution, and that $1-x \le e^{-x}$.

Using the union bound, we get

$$\begin{aligned}
\Pr\left[\bigcup_{j\in I} S_j \ne E\right] &\le \sum_{i=1}^{n}\Pr\left[e_i \text{ is not covered by } \bigcup_{j\in I} S_j\right] \\
&\le n\frac{1}{n^2} = \frac{1}{n}.
\end{aligned}$$

□

Remark 1 *Notice that Lemma 1 bounds the value of the returned solution but we're only interested in that bound if the solution is a valid one. So what we are really interested in bounding is the quantity:*

$$\mathbb{E}\left[\sum_{j\in I} w_j \,\middle|\, I \text{ is a set cover}\right]$$

Using the inequality $\mathbb{E}[X|E] \le \mathbb{E}[X]/\Pr[E]$ *for non-negative random variables* X*, one can obtain a similar bound to Lemma* 1 *for the conditional expectation with only an extra factor of* 2 *on the right-hand side which is still a* $O(\log n)$ *approximation as promised (see [2, Theorem 1.17] for more details).*

2 Online Set Cover

In the *online* version of the problem, we assume we know the sets $E, S_1, S_2, \ldots, S_m$ along with the corresponding weights $w_1, \ldots, w_m$ at time zero. At each time-step i, an element e_i arrives[1] and we must update the current solution by (potentially) adding one or more sets S_j such that e_i is covered. The goal is to come up with an algorithm which obtains a

[1]The order in which elements arrive is unknown to the algorithm and we can assume it is decided by an adversary.

good *competitive ratio*, defined as follows. For each time step i, let E' be the set of elements having arrived so far. We say the competitive ratio of an algorithm is α if the cost of the online solution is at most α times the cost of the optimal solution for E' and this holds for *every* time-step i.

The idea is to try and solve the corresponding LP in an online manner and then use this solution to guide our choice of which set to include in the online solution.

2.1 Online LP Solver

The first step in our plan is to come up with a (deterministic) online algorithm for maintaining a solution x^* such that:

$$\begin{aligned} &\textstyle\sum_{j:e_i \in S_j} x_j^* \geq 1, \quad \forall e_i \in E' \\ &x_j^* \geq 0 \end{aligned}$$

Notice that whenever a new element arrives, a new constraint is added to the primal LP and a new variable is added to the dual. We're going to update the online solution in a non-decreasing manner so that previous constraints are still satisfied after each update. Eventually, we want to show that for each time-step:

$$\sum_{j=1}^{m} w_j \cdot x_j^* \leq O(\log d) \cdot (\text{optimal LP solution for } E')$$

where d is defined as follows: let $f_i = |\{j \ : \ e_i \in S_j\}|$, then $d = \max_i f_i$.

In what follows we are going to make the assumption that $w_j \geq 1$ for all j[2]. Consider Algorithm 2 which resembles the multiplicative weights update ideas we've seen in previous lectures.

Algorithm 2: Online LP Solver

$x_j \leftarrow 0, \ j = 1, \ldots, m$
$y_i \leftarrow 0, \ i = 1, \ldots, n$
while *new elements* e_i *arrives* **do**
  while $\sum_{j:e_i \in S_j} x_j < 1$ **do**
    $y_i \leftarrow y_i + 1$
    for *each* j *such that* $e_i \in S_j$ **do**
      $x_j \leftarrow x_j \left(1 + \frac{1}{w_j}\right) + \frac{1}{f_i \cdot w_j}$
    end
  end
end

Here are three useful lemmas regarding the performance of Algorithm 2, the proof of which we'll present shortly.

Lemma 3 *Algorithm 2 produces a feasible solution.*

[2]If that is not the case we can adjust the weights by scaling all of them by the same value.

Lemma 4 *In each iteration of the outer while loop of Algorithm 2:*

$$\textit{(increase in cost of primal)} \leq 2 \cdot \textit{(increase in cost of dual)}$$

Lemma 5 *Let $\vec{y}$ be a dual solution maintained at some time-step i in Algorithm 2. Then the following scaled version is a feasible dual solution:*

$$\frac{\vec{y}}{2\ln(3d+1)}$$

Using the above results we're able to prove now the following Theorem.

Theorem 6 *Algorithm 2 is $O(\log n)$-competitive.*

Proof: Consider a time-step where a set E' of elements have arrived and denote by LP_{OPT} the optimal LP solution to the partial LP (containing only the constrains related to elements in E').

First, notice that by repeated application of Lemma 4 we get the following inequality on the current values of x_i, y_i maintained by the algorithms:

$$\sum_{j=1}^{m} w_j x_j \leq 2 \sum_{i=1}^{n} y_i. \tag{1}$$

On the other hand, by Lemma 5 we can get the following inequality:

$$\sum_{i=1}^{n} y_i \cdot \frac{1}{2\ln(3d+1)} \leq \text{LP}_{\text{OPT}}, \text{ or} \tag{2}$$

$$\sum_{i=1}^{n} y_i \leq O(\log d) \cdot \text{LP}_{\text{OPT}}. \tag{3}$$

Combining inequalities (1) and (3) and the fact that x is a feasible solution (by Lemma 3), we derive the theorem:

$$\sum_{j=1}^{m} w_j x_j \leq O(\log n) \cdot \text{LP}_{\text{OPT}}.$$

□

We now proceed to prove Lemmas 4 and 5 — Lemma 3 trivially holds by construction.

Proof of Lemma 4: In each iteration of the inner while loop of Algorithm 2, the dual objective is increased by exactly 1 while the primal objective is increased by:

$$\sum_{j:e_i \in S_j} w_j \underbrace{\left(\frac{x_j}{w_j} + \frac{1}{f_i w_j}\right)}_{x_j^{\text{new}} - x_j^{\text{old}}} = \sum_{j:e_i \in S_j} \left(x_j + \frac{1}{f_i}\right) \leq 1 + f_i \cdot \frac{1}{f_i} = 2$$

where the inequality follows from the feasibility of x. □

Proof of Lemma 5: We need to show that the proposed scaled dual solution satisfies all dual constraints: $\sum_{i:e_i \in S_j} y_i/(2\ln(3d+1)) \leq w_j$. To this end, we're going to follow a strategy similar to the one used in proving guarantees for the performance of the multiplicative weights update algorithms. Specifically, we are going to upper and lower bound x_j and then combine those bounds to get the desired upper bound on the dual objective.

The first claim is that x_j never gets too large and in particular $x_j \leq 3$ in each iteration of the algorithm. Indeed, if x_j is ever greater than 1 it is never increased again so the final value of x_j is bounded by:

$$x_j^{\text{final}} = x_j^{\text{prev}} \left(1 + \frac{1}{w_j}\right) + \frac{1}{f_i w_j} \leq 1 + \frac{1}{w_j} + \frac{1}{f_i w_j} \leq 1 + 1 + 1 = 3, \tag{4}$$

where the last inequality follows by the assumption that all $w_j \geq 1$ and $f_i \geq 1$ which is made without loss of generality.

On the other hand, we can prove by induction the following inequality at each time-step:

$$x_j \geq \frac{1}{d}\left(\left(1 + \frac{1}{w_j}\right)^{\sum_{i:e_i \in S_j} y_i} - 1\right). \tag{5}$$

Before proving this, let's see how the Lemma follows by combining Inequalities (4) and (5):

$$\begin{aligned}
\frac{1}{d}\left(\left(1 + \frac{1}{w_j}\right)^{\sum_{i:e_i \in S_j} y_i} - 1\right) &\leq 3 \\
\left(1 + \frac{1}{w_j}\right)^{\sum_{i:e_i \in S_j} y_i} &\leq 3d + 1 \\
\left(\sum_{i:e_i \in S_j} y_i\right) \ln\left(1 + \frac{1}{w_j}\right) &\leq \ln(3d+1) \\
\left(\sum_{i:e_i \in S_j} y_j\right) \frac{\frac{1}{w_j}}{1 + \frac{1}{w_j}} &\leq \ln(3d+1) \\
\frac{1}{w_j + 1} \sum_{i:e_i \in S_j} y_i &\leq \ln(3d+1) \\
\sum_{i:e_i \in S_j} \frac{y_i}{2\ln(3d+1)} &\leq w_j
\end{aligned}$$

where we used the inequality $\ln(1+x) \geq \frac{x}{1+x}$ and $w_j \geq 1$.

We now prove by induction on the number of iterations of the inner while loop (through the run of the algorithm) that Inequality (5) holds.

For the base of the induction notice that $y_i = 0$ initially and the inequality reduces to $x_j \geq 0$ which holds true. Suppose now that the inequality holds at the end of an iteration where the currently maintained primal solution is x_j and the corresponding dual is y_j. On the next iteration, y_{i^*} is increased by 1 for some element i^* to give a updated dual vector

y' and some of the x_j are updated to values x'_j to give rise to an updated primal vector x' for which the following hold:

$$\begin{aligned}
x'_j &= x_j\left(1+\frac{1}{w_j}\right)+\frac{1}{f_i w_j} \\
&\geq x_j\left(1+\frac{1}{w_j}\right)+\frac{1}{d}\frac{1}{w_j} && \text{(by definition of } d\text{)} \\
&\geq \frac{1}{d}\left(\left(1+\frac{1}{w_j}\right)^{\sum_{i:e_i\in S_j} y_i}-1\right)\left(1+\frac{1}{w_j}\right)+\frac{1}{dw_j} && \text{(induction hypothesis)} \\
&\geq \frac{1}{d}\left(\left(1+\frac{1}{w_j}\right)^{\sum_{i:e_i\in S_j} y'_i}-1\right).
\end{aligned}$$

□

2.2 Algorithm for the Online Set Cover Problem

The algorithm presented in the previous section only solves the Online Linear Program related to the Set Cover Problem. We now present Algorithm 3 which we'll prove is $O(\log n \cdot \log d)$-competitive for the Online Set Cover problem.

Algorithm 3: Online Set Cover Algorithm

```
I ← ∅
n ← 0
while a new element e_i arrives do
    n ← n + 1
    Run an iteration of Algorithm 2 for the newly arrived element to get solution x*
    for j ← 1 to m do
        Pick more (if necessary) X(j,k) ∈ [0,1] uniformly at random in order to
         have enough for k to range from 1 up to 2 ln n
        T_j ← min_k X(j,k)
        if x*_j ≥ T_j then
            I ← I ∪ {j}
        end
    end
end
```

Theorem 7 *Algorithm 3 is $O(\log n \log d)$-competitive.*

Proof: The crucial fact to notice here is that every x^*_j of the current LP solution is non-decreasing in time by construction of the online LP solver and T_j is non-increasing in time. Consequently, if the inequality $x^*_j \geq T_j$ holds at some time-step then it continues to hold for every subsequent time-step.

This allows essentially the same proof we did in Lemma 1 to go through and get that for every time-step:

$$\mathbb{E}\left[\sum_{j\in I} w_j\right] \leq (2\ln n)\sum_{j=1}^{m} w_j x_j^* \leq O(\log n \log d)\mathrm{LP}_{\mathrm{OPT}} \leq O(\log n \log d)\,\mathrm{OPT}\,.$$

where in the second-to-last step we used Theorem 6.

A similar argument to the one presented in the proof of Lemma 2 can bound the probability of returning an invalid cover and conclude that the competitiveness guarantee of Algorithm 3 holds with high probability. □

References

[1] Niv Buchbinder and Joseph Naor. *The Design of Competitive Online Algorithms via a Primal-Dual Approach.* Now Publishers Inc., Hanover, MA, USA, 2009.

[2] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms.* Cambridge University Press, New York, NY, USA, 1st edition, 2011.