

Notes edited by instructor in 2011.

We introduce the use linear programming (LP) in the design and analysis of approximation algorithms. The topics include Vertex Cover, Set Cover, randomized rounding, dual-fitting. It is assumed that the students have some background knowledge in basics of linear programming.

1 Vertex Cover via LP

Let $G = (V, E)$ be an undirected graph with arc weights $w : V \rightarrow R^+$. Recall the vertex cover problem from previous lecture. We can formulate it as an integer linear programming problem as follows. For each vertex v we have a variable x_v . We interpret the variable as follows: if $x_v = 1$ if v is chosen to be included in a vertex cover, otherwise $x_v = 0$. With this interpretation we can easily see that the minimum weight vertex cover can be formulated as the following integer linear program.

$$\begin{aligned} \min \quad & \sum_{v \in V} w_v x_v \\ \text{subject to} \quad & \\ & x_u + x_v \geq 1 \quad \forall e = (u, v) \in E \\ & x_v \in \{0, 1\} \quad \forall v \in V \end{aligned}$$

However, solving integer linear programs is NP-Hard. Therefore we use Linear Programming (LP) to approximate the optimal solution, $\text{OPT}(I)$, for the integer program. First, we can relax the constraint $x_v \in \{0, 1\}$ to $x_v \in [0, 1]$. It can be further simplified to $x_v \geq 0, \forall v \in V$.

Thus, a linear programming formulation for Vertex Cover is:

$$\begin{aligned} \min \quad & \sum_{v \in V} w_v x_v \\ \text{subject to} \quad & \\ & x_u + x_v \geq 1 \quad \forall e = (u, v) \in E \\ & x_v \geq 0 \end{aligned}$$

We now use the following algorithm:

VERTEX COVER VIA LP:
Solve LP to obtain an optimal fractional solution x^*
Let $S = \{v \mid x_v^* \geq \frac{1}{2}\}$
Output S

Then the following claims are true:

Claim 1 S is a vertex cover.

Proof: Consider any edge, $e = (u, v)$. By feasibility of x^* , $x_u^* + x_v^* \geq 1$, and thus either $x_u^* \geq \frac{1}{2}$ or $x_v^* \geq \frac{1}{2}$. Therefore, at least one of u and v will be in S . \square

Claim 2 $w(S) \leq 2\text{OPT}_{LP}(I)$.

Proof: $\text{OPT}_{LP}(I) = \sum_v w_v x_v^* \geq \frac{1}{2} \sum_{v \in S} w_v = \frac{1}{2} w(S)$ \square

Therefore, $\text{OPT}_{LP}(I) \geq \frac{\text{OPT}(I)}{2}$ for all instances I .

Note: For minimization problems: $\text{OPT}_{LP}(I) \leq \text{OPT}(I)$, where $\text{OPT}_{LP}(I)$ is the optimal solution found by LP; for maximization problems, $\text{OPT}_{LP}(I) \geq \text{OPT}(I)$.

Integrality Gap

We introduce the notion of *integrality gap* to show the best approximation guarantee we can acquire by using the LP optimum as a lower bound.

Definition: For a minimization problem Π , the integrality gap for a linear programming relaxation/formulation LP for Π is $\sup_{I \in \Pi} \frac{\text{OPT}(I)}{\text{OPT}_{LP}(I)}$.

That is, the integrality gap is the worst case ratio, over all instances I of Π , of the integral optimal value and the fractional optimal value. Note that different linear programming formulations for the same problem may have different integrality gaps.

Claims 1 and 2 show that the integrality gap of the Vertex Cover LP formulation above is at most 2.

Question: Is this bound tight for the Vertex Cover LP?

Consider the following example: Take a complete graph, K_n , with n vertices, and each vertex has $w_v = 1$. It is clear that we have to choose $n - 1$ vertices to cover all the edges. Thus, $\text{OPT}(K_n) = n - 1$. However, $x_v = \frac{1}{2}$ for each v is a feasible solution to the LP, which has a total weight of $\frac{n}{2}$. So gap is $2 - \frac{1}{n}$, which tends to 2 as $n \rightarrow \infty$.

Other Results on Vertex Cover

1. The current best approximation ratio for Vertex Cover is $2 - \Theta(\frac{1}{\sqrt{\log n}})$ [1].
2. Open problem: obtain a $2 - \epsilon$ approximation or to prove that it is NP-hard to obtain $2 - \epsilon$ for any fixed $\epsilon > 0$. Current best hardness of approximation: unless $P=NP$, there is no 1.36 approximation for Vertex Cover [2].
3. The vertex cover problem can be solved optimally in polynomial time for bipartite graphs. This follows from what is known as Kőnig's theorem.
4. The vertex cover problem admits a polynomial time approximation scheme (PTAS), that is a $(1 + \epsilon)$ -approximation for any fixed $\epsilon > 0$, for planar graphs. This follows from a general approach due to Baker [?].

2 Set Cover via LP

The input to the Set Cover problem consists of a finite set $U = \{1, 2, \dots, n\}$, and m subsets of U , S_1, S_2, \dots, S_m . Each set S_j has a non-negative weight w_j and the goal is to find the minimum weight collection of sets which cover all elements in U (in other words their union is U).

A linear programming relaxation for Set Cover is:

$$\begin{aligned} \min \quad & \sum_j w_j x_j \\ \text{subject to} \quad & \\ & \sum_{j:i \in S_j} x_j \geq 1 \quad \forall i \in \{1, 2, \dots, n\} \\ & x_j \geq 0 \quad 1 \leq j \leq m \end{aligned}$$

And its dual is:

$$\begin{aligned} \max \quad & \sum_{i=1}^n y_i \\ \text{subject to} \quad & \\ & \sum_{i \in S_j} y_i \leq w_j \quad \forall j \in \{1, 2, \dots, m\} \\ & y_i \geq 0 \quad \forall i \in \{1, 2, \dots, n\} \end{aligned}$$

We give several algorithms for Set Cover based on this primal/dual pair LPs.

2.1 Deterministic Rounding

SET COVER VIA LP:

Solve LP to obtain an optimal solution x^* , which contains fractional numbers.

Let $P = \{i \mid x_i^* \geq \frac{1}{f}\}$, where f is the maximum number of sets that contain any element

Output $\{S_j \mid j \in P\}$

Note that the above algorithm, even when specialized to vertex cover, is different from the one we saw earlier. It includes all sets which have a strictly positive value in an *optimum* solution the LP.

Let x^* be an optimal solution to the primal LP, y^* be an optimum solution to the dual, and let $P = \{j \mid x_j^* > 0\}$. First, note that by strong duality, $\sum_j w_j x_j^* = \sum_i y_i^*$. Second, by complementary slackness if $x_j^* > 0$ then the corresponding dual constraint is tight, that is $\sum_{i \in S_j} y_i^* = w_j$.

Claim 3 *The output of the algorithm is a feasible set cover for the given instance.*

Proof: Exercise. □

Claim 4 $\sum_{j \in P} w_j \leq f \sum_j w_j x_j^* = \text{OPT}_{LP}$.

Proof:

$$\sum_{j \in P} w_j = \sum_{j: x_j^* > 0} (w_j) = \sum_{j: x_j^* > 0} \left(\sum_{i \in S_j} y_i^* \right) = \sum_i y_i^* \left(\sum_{j: i \in S_j, x_j^* > 0} 1 \right) \leq f \sum_i y_i^* \leq f \text{OPT}_{LP}(I).$$

□

Notice that the the second equality is due to complementary slackness conditions (if $x_j > 0$, the corresponding dual constraint is tight), the penultimate inequality uses the definition of f , and the last inequality follows from weak duality (a feasible solution for the dual problem is a lower bound on the optimal primal solution).

Therefore we have that the algorithm outputs a cover of weight at most $f \text{OPT}_{LP}$. We note that f can be as large as n in which case the bound given by the algorithm is quite weak. In fact, it is not construct examples that demonstrate the tightness of the analysis.

Remark: The analysis crucially uses the fact that x^* is an optimal solution. On the other hand the algorithm for vertex cover is more robust.

2.2 Randomized Rounding

Now we describe a different rounding that yields an approximation bound that does not depend on f .

SOLVING SET COVER VIA RANDOMIZED ROUNDING:
 $A = \emptyset$, and let x^* be an optimal solution to the LP.
for $k = 1$ to $2 \ln n$ do
 pick each S_j independently with probability x_j^*
 if S_j is picked, $A = A \cup \{j\}$
end for
Output the sets with indices in A

Claim 5 $Pr[i \text{ is not covered in an iteration}] = \prod_{j: i \in S_j} (1 - x_j^*) \leq \frac{1}{e}$.

Intuition: We know that $\sum_{j: i \in S_j} x_j^* \geq 1$. Subject to this constraint, if and want to minimize the probability, we can let x_j^* equal to each other, then the probability $= (1 - \frac{1}{k})^k$, where $x_j^* = 1/k$.

Proof: $Pr[i \text{ is not covered in an iteration}] = \prod_{j: i \in S_j} (1 - x_j^*) \leq \prod_{j: i \in S_j} e^{-x_j^*} \leq e^{-\sum_{j: i \in S_j} x_j^*} \leq \frac{1}{e}$.
□

We then obtain the following corollaries:

Corollary: $Pr[i \text{ is not covered at the end of the algorithm}] \leq e^{-2 \log n} \leq \frac{1}{n^2}$.

Corollary: $Pr[\text{all elements are covered, after the algorithm stops}] \geq 1 - \frac{1}{n}$. The above follows from the union bound. The probability that i is not covered is at most $1/n^2$, hence the probability that there is some i that is not covered is at most $n \cdot 1/n^2 \leq 1/n$.

Let $C_t =$ cost of sets picked in iteration t , then $E[C_t] = \sum_{j=1}^m w_j x_j^*$, where $E[X]$ denotes the expectation of a random variable X . Then, let $C = \sum_{t=1}^{2 \ln n} C_t$; we have $E[C] = \sum_{t=1}^{2 \ln n} E[C_t] \leq 2 \ln n \text{OPT}_{LP}$. We know that $\Pr[C > 2E[C]] \leq \frac{1}{2}$ by Markov's inequality, so we have $\Pr[C \leq 4 \ln n \text{OPT}_{LP}] \geq \frac{1}{2}$. Therefore, $\Pr[C \leq 4 \ln n \text{OPT}_{LP}$ and all items are covered] $\geq \frac{1}{2} - \frac{1}{n}$. Thus, the randomized rounding algorithm, with probability close to $1/2$ succeeds in giving a feasible solution of cost $O(\log n) \text{OPT}_{LP}$. Note that we can check whether the solution satisfies the desired properties (feasibility and cost) and repeat the algorithm if it does not.

1. We can check if solution after rounding satisfies the desired properties, such as all elements are covered, or cost at most $2c \log n \text{OPT}_{LP}$. If not, repeat rounding. Expected number of iterations to succeed is a constant.
2. We can also use Chernoff bounds (large deviation bounds) to show that a single rounding succeeds with high probability (probability at least $1 - \frac{1}{\text{poly}(n)}$).
3. The algorithm can be *derandomized*. Derandomization is a technique of removing randomness or using as little randomness as possible. There are many derandomization techniques, such as the method of conditional expectation, discrepancy theory, and expander graphs. Broder *et al.* [5] use min-wise independent permutations to derandomize the RNC algorithm for approximate set cover due to S. Rajagopalan and V. Vazirani [6].
4. After a few rounds, select the cheapest set that covers each uncovered element. This has low expected cost. This algorithm ensures feasibility but guarantees cost only in the expected sense.

Other Results related to Set Cover

1. Unless $P = NP$, there is no $c \log n$ approximation for some fixed c [4].
2. Unless $NP \subseteq \text{DTIME}(n^{O(\log \log n)})$, there is no $(1 - o(1)) \ln n$ -approximation [3].
3. Unless $P = NP$, there is no $(1 - \frac{1}{e} + \varepsilon)$ -approximation for max-coverage for any fixed $\varepsilon > 0$.

2.3 Dual-fitting

In this section, we introduce the technique of dual-fitting for the analysis of approximation algorithms. At a high-level the approach is the following:

1. Construct a feasible solution to the dual LP.
2. Show that the cost of the solution returned by the algorithm can be bounded in terms of the value of the dual solution.

Note that the algorithm itself need not be LP based. Here, we use Set Cover as an example. Please refer to the previous section for the primal and dual LP formulations of Set Cover.

We can interpret the dual as follows: Think of y_i as how much element i is willing to pay to be covered; the dual maximizes the total payment, subject to the constraint that for each set, the total payment of elements in that set is at most the cost of the set.

The greedy algorithm for weighted Set Cover is as follows:

GREEDY SET COVER:

$Covered = \emptyset;$
 $A = \emptyset;$
 While $Covered \neq U$ do
 $j \leftarrow \arg \min_k (\frac{w_k}{|S_k \cap Uncovered|});$
 $Covered = Covered \cup S_j;$
 $A = A \cup \{j\}.$
 end while;
 Output sets in A as cover

Theorem 6 GREEDY SET COVER picks a solution of cost $\leq H_d \cdot \text{OPT}_{LP}$, where d is the maximum set size, i.e., $d = \max_j |S_j|$.

To prove this, we can augment the algorithm a little bit:

AUGMENTED GREEDY ALGORITHM OF WEIGHTED SET COVER:

$Covered = \emptyset;$
 while $Covered \neq U$ do
 $j \leftarrow \arg \min_k (\frac{w_k}{|S_k \cap Uncovered|});$
 if i is uncovered and $i \in S_j$, set $p_i = \frac{w_j}{|S_j \cap Uncovered|};$
 $Covered = Covered \cup S_j;$
 $A = A \cup \{j\}.$
 end while;
 Output sets in A as cover

It is easy to see that the algorithm outputs a set cover.

Claim 7 $\sum_{j \in A} w_j = \sum_i p_i$.

Proof: Consider when j is added to A . Let $S'_j \subseteq S_j$ be the elements that are uncovered before j is added. For each $i \in S'_j$ the algorithm sets $p_i = w_j / |S'_j|$. Hence, $\sum_{i \in S'_j} p_i = w_j$. Moreover, it is easy to see that the sets $S'_j, j \in A$ are disjoint and together partition U . Therefore,

$$\sum_{j \in A} w_j = \sum_{j \in A} \sum_{i \in S'_j} p_i = \sum_{i \in U} p_i.$$

□

For each i , let $y'_i = \frac{1}{H_d} p_i$.

Claim 8 y' is a feasible solution for the dual LP.

Suppose the claim is true, then the cost of GREEDY SET COVER's solution $= \sum_i p_i = H_d \sum_i y'_i \leq H_d \text{OPT}_{LP}$. The last step is because any feasible solution for the dual problem is a lower bound on the value of the primal LP (weak duality).

Now, we prove the claim. Let S_j be an arbitrary set, and let $|S_j| = t \leq d$. Let $S_j = \{i_1, i_2, \dots, i_t\}$, where we the elements are ordered such that i_1 is covered by Greedy no-later than i_2 , and i_2 is covered no later than i_3 and so on.

Claim 9 For $1 \leq h \leq t$, $p_{i_h} \leq \frac{w_j}{t-h+1}$.

Proof: Let $S_{j'}$ be the set that covers i_h in Greedy. When Greedy picked $S_{j'}$ the elements i_h, i_{h+1}, \dots, i_t from S_j were uncovered and hence Greedy could have picked S_j as well. This implies that the density of $S_{j'}$ when it was picked was no more than $\frac{w_j}{t-h+1}$. Therefore p_{i_h} which is set to the density of $S_{j'}$ is at most $\frac{w_j}{t-h+1}$. \square

From the above claim, we have

$$\sum_{1 \leq h \leq t} p_{i_h} \leq \sum_{1 \leq h \leq t} \frac{w_j}{t-h+1} = w_j H_t \leq w_j H_d.$$

Thus, the setting of y'_i to be p_i scaled down by a factor of H_d gives a feasible solution.

References

- [1] G. Karakostas. A better approximation ratio for the Vertex Cover problem. *ECCC Report TR04-084*, 2004.
- [2] I. Dinur and S. Safra. The importance of being biased. *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 33-42, 2002.
- [3] U. Feige. A Threshold of $\ln n$ for Approximating Set Cover. *Journal of the ACM*, v.45 n.4, p.634 - 652, 1998.
- [4] R. Raz and M. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. *Proceedings of STOC 1997*, pp. 475-484, 1997.
- [5] A. Z. Broder, M. Charikar, and M. Mitzenmacher. A derandomization using min-wise independent permutations *Journal of Discrete Algorithms*, Volume 1, Issue 1, pages 11-20, 2003.
- [6] S. Rajagopalan and V. Vazirani. Primal-Dual RNC Approximation Algorithms for Set Cover and Covering Integer Programs *SIAM Journal on Computing*, Volume 28, Issue 2, p.525 - 540, 1999.