

In lectures 9 and 10, we saw how to use Linear Programming (LP) to approximate problems or use it as a technique to analyze an algorithm. The topics include Vertex Cover, Set Cover, Randomized Rounding, Dual-Fitting, and Totally Unimodular Matrices.

Note: For more reviews of Linear Programming, please refer to the old slides or the links on the website.

1 Vertex Cover Via LP

Let $G = (V, E)$ be an undirected graph with arc weights $w : V \rightarrow R^+$. Define x_v for each vertex v as follows: $x_v = 1$, if v is in the vertex cover; $x_v = 0$, if v is not chosen. Our goal is to find $\min(\sum_{v \in V} w_v x_v)$, such that $x_u + x_v \geq 1, \forall e = (u, v) \in E, x_v \in \{0, 1\}$.

However, we can't solve Integer Linear Programming (ILP) problems in polynomial time. So we have to use Linear Programming (LP) to approximate the optimal solution, $\text{OPT}(I)$, produced by ILP. 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:

SOLVE VERTEX COVER VIA LP:
Solve LP to obtain an optimal solution x^* , which contains fractional numbers.
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 . □

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

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

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 LP as a lower bound.

Definition: For a problem Π , the integrality gap for a linear program LP 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 , between 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, if we let each $x_v = \frac{1}{2}$, then this is one feasible solution to the LP, which gives a total weight of $\frac{n}{2}$. So gap is $2 - \frac{1}{n}$, which tends to 2 as n tends to infinity. Hence, this bound is tight.

Other Results on Vertex Cover

1. The current best approximation ratio for Vertex Cover is $2 - \Theta(\frac{1}{\sqrt{\log n}})(2 - O(1))$ [1].
2. Open problem: obtain a $2 - \varepsilon$ approximation or to prove that it is NP-hard to obtain $2 - \varepsilon$ for any fixed $\varepsilon > 0$
3. Current best hardness of approximation: unless $P=NP$, there is no 1.36 approximation for Vertex Cover [2].

2 Set Cover Via LP

Given a universe, $U = \{1, 2, \dots, n\}$, and m subsets of U , s_1, s_2, \dots, s_m , with $w_j =$ weight of set s_j , our goal is to find the minimum weight collection of sets which cover all elements in U .

Thus, a Linear Program for Set Cover is:

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

And its dual is:

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

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

SOLVING SET COVER VIA LP:

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

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

Output S

Let x^* be an optimal solution to the primal LP, y^* be the corresponding dual solution, and let $S = \{j \mid x_j^* > 0\}$

Claim 3 $w(S) \leq f \sum w_j x_j^*$

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

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).

Randomized Rounding for Set Cover

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 A

Claim 4 $Pr[i \text{ is not covered in an iteration}] = \prod_{j: i \in s_j} (1 - x_j^*) \leq \frac{1}{e}$.

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: $Pr[\text{all elements are covered, after the algorithm stops}] \geq 1 - \frac{1}{n}$.

Analysis

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, $Pr[C \leq 4 \ln n \text{OPT}_{LP} \text{ and all items are covered}] \geq \frac{1}{2} - \frac{1}{n}$. Thus, the randomized rounding algorithm for Set Cover is an $O(\log n)$ -approximation.

Fixing

Note that the algorithm above only gives an good approximation with some probability, so we need to do some extra work to fix the solution.

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.* [6] use min-wise independent permutations to derandomize the RNC algorithm for approximate set cover due to S. Rajagopalan and V. Vazirani [7].
4. After a few rounds, select the cheapest set that covers each uncovered element. This has low expected cost.

Other Results related to Set Cover

1. Unless $P = NP$, there is no $c \log n$ approximation for some fixed c [4].
2. Unless $NP \subseteq DTIME(n^{O(\log \log n)})$, 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 $\varepsilon > 0$.

3 Dual-fitting

In this section, we introduce the technique of dual-fitting, to use duality for the analysis of approximation algorithms.

The standard procedure of dual-fitting is:

1. While executing the algorithm, 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.

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; our goal is to maximize 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:
 $Covered = \emptyset$;
while $Covered \neq U$ do
 $j \leftarrow \arg \min_k (\frac{w_k}{|s_k \cap Uncovered|})$;
 $Covered = Covered \cup s_j$;
 Add j to set cover;
end while;

Theorem 5 GREEDY SET COVER picks a solution of cost $\leq H_d \cdot \text{OPT}_{LP}$, where d is the maximum set size, i.e., $d = \max_j |s_j|$.

Proof: 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 $p_i = \frac{w_j}{|s_j \cap Uncovered|}$;
 $Covered = Covered \cup s_j$;
 Add j to set cover;
end while;

Since every element i is covered by any solution, the cost of the solution returned by GREEDY SET COVER = $\sum_j p_j$, for all $j \in U$.

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

Claim 6 y' is a feasible solution for the dual problem.

Suppose the claim is true, then the cost of GREEDY SET COVER's solution = $\sum_j p_j = H_d \sum_j y'_j \leq H_d \text{OPT}_{LP}$. 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 start to prove the claim. Let s_j be an arbitrary set, and let $|s_j| = t \leq d$. So we can denote s_j as $\{j_1, j_2, \dots, j_t\}$, where j_i are ordered in the way that j_1 is picked before j_2 , j_2 is picked before j_3 , and so on.

Claim 7 For $1 \leq h \leq t$, $p_{j_h} \leq \frac{w_j}{t-h+1}$

This is because when j_h was covered, Greedy could have picked s_j at density $\frac{w_j}{t-h+1}$. With this claim, we know that $\sum_{1 \leq h \leq t} p_{j_h} \leq w_j H_t \leq w_j H_d$.

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

Question: Is this bound tight?

Consider the following example: Pick a set of n elements, and $10 \log n$ sets. For each element, i , randomly pick $5 \log n$ sets, and assign i to the chosen sets. If every set has the value $\frac{1}{5 \log n}$, then this is a feasible fractional solution. Thus, the total cost is $\frac{10 \log n}{5 \log n} = 2$. On the other hand, consider a fixed collection of $\frac{\log n}{2}$ sets. The probability that these sets will cover all elements is quite small (It's $\approx e^{-\sqrt{n}}$). Therefore, using the union bound, the probability that *any* collection of $\frac{\log n}{2}$ sets will cover all elements is small. Thus, the integrality gap of the LP is $\Omega(\log n)$.

4 Totally Unimodular Matrices

We are interested in TUM matrices, because if an Integer Linear Program's constraint matrix is Totally Unimodular, and has an integer vector on the right hand side, then this ILP can be solved by Linear Programming since all its basic feasible solutions are integral.

Integral Polyhedra

Before discussing about TUM matrices, we first introduce the term Integral Polyhedra.

Definition: A rational polyhedron given by a system of inequalities $Ax \leq b$ is integral if and only if all its vertices have integer coordinates.

Theorem 8 *The polyhedron, $P : Ax \leq b$, is integral if and only if for each integral w , the optimum value of $\max(w x)$, such that $Ax \leq b$ is an integer if it is finite [5].*

Totally Unimodular Matrices

Definition: A $m \times n$ matrix A is said to be totally unimodular if every square submatrix A' of A has the property that $\det(A') \in \{-1, 0, 1\}$.

Claim 9 *If A is TUM and b is an integer vector, then the polyhedron $Ax \leq b$ is an integral polyhedron*

Proof: If v is a vertex of the polyhedron $Ax \leq b$, then there is a square submatrix A' and a vector b' such that v is the solution to $A'x = b'$. Thus, $v = A'^{-1}b' = \frac{1}{\det(A')} \text{adj}(A')b'$, where $\text{adj}(A')$ denotes the adjoint of matrix A' . Since A is TUM, $\det(A') \in \{1, -1\}$. Therefore, v is integral, since both $\text{adj}(A')$ and b' are integral. \square

Preserving total unimodularity

There are some operations that preserve total unimodularity.

1. We can add box constraints to $Ax \leq b$: The system $Ax \leq b, \ell \leq x \leq u$ for integer b, ℓ, u is an integral polyhedron if A is TUM.
2. The dual of $\max cx, Ax \leq b$ is $\min yb, yA^t \geq c$: If c, b are integer, then both primal and dual are integer polyhedra.

Examples

In this subsection, we will introduce some examples to show that why we are interested in TUM matrices.

Theorem 10 *Let $G = (V, E)$ be a directed graph, and let A_G be its arc-vertex incidence matrix. Then A_G is TUM.*

Proof: Let A_m be a $m \times m$ submatrix of A_G . Proof by induction on m .

Base case, when $m = 1$: Since every element in A_G is either 1, 0, or -1, A_1 must be TUM.

Induction hypothesis: assume for $m \leq k$, A_m is TUM. Then we need to consider three cases for A_{m+1} :

Case 1: A_{m+1} has a zero column. Then $\det(A_{m+1}) = 0$, so A_{m+1} is TUM.

Case 2: A_{m+1} has a column with exactly one 1 or -1, and all other entries are 0. Calculating $\det(A_{m+1})$ using this column. From the induction hypothesis, A_{m+1} is TUM.

Case 3: Every column of A_{m+1} has one 1 and one -1. Since the sum of A'_{m+1} 's row vectors is a zero vector, $\det(A_{m+1}) = 0$, and thus A_{m+1} is TUM. \square

Note: This explains why we can solve flow problems on directed graphs in polynomial time.

Theorem 11 *$G = (V, E)$ is a bipartite graph \iff the arc-vertex incidence matrix A_G is TUM.*

Proof: " \Leftarrow ": Let A_G be a totally unimodular matrix. Assume G is not bipartite, then G contains an odd cycle. A_G 's submatrix, which corresponds to the odd cycle has determinant of 2. This contradicts the total unimodularity of A .

" \Rightarrow ": Let G be a bipartite graph, and construct a new graph $G' = (X \cup Y, E')$, where X is on one side, Y is on the other side, and E' represents directed edges from X to Y . Let $A_{G'}$ be the incidence matrix of G' , where $A(u, e) = 1$ and $A(v, e) = -1$, if $e = (u, v) \in E'$. From the theorem above, $A_{G'}$ is TUM. Notice that after the construction, $\det(A_G)$ and $\det(A_{G'})$ will differ from each other by a multiplicative factor of 1 or -1, i.e., $\det(A_G) = \pm \det(A_{G'})$. Hence, A_G is also TUM. \square

Note: This explains why we can solve the matching problem on bipartite graphs in polynomial time.

Theorem 12 *If A is a 0, 1 matrix, and has the consecutive 1's property, then A is TUM. We say that a matrix has the consecutive 1's property, if the rows of the matrix can be permuted so that the 1's in each column appear consecutively..*

Proof Sketch. Let B be an $m \times m$ submatrix of A , and assume that any smaller submatrix of B has determinant in $\{0, 1, -1\}$. Use induction on the number of 1s in the first row: If 0, the determinant is 0; if 1, the determinant is ± 1 times the determinant of a submatrix of B . Otherwise, let c_1, c_2 be two columns with 1s in the first row, and let c_1 have at least as many 1s as c_2 . Now, replace c_1 by $c_1 - c_2$; this does not change the determinant, but there are fewer 1s in the first row. \square

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] J. Edmonds and R. Giles. A min-max relation for submodular functions on graphs. *In Studies in Integer Programming, Annals of Discrete Mathematics 1*, pages 185V204, 1977.
- [6] 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.
- [7] J. Edmonds and R. Giles. Primal-Dual RNC Approximation Algorithms for Set Cover and Covering Integer Programs *SIAM Journal on Computing*, Volume 28, Issue 2, p.525 - 540, 1999.