COMPSCI 632: Approximation Algorithms September 18, 2017

Lecture 7

Lecturer: Debmalya Panigrahi *Scribe: Xiang Wang*

1 Overview

In this lecture, we will use Primal-Dual method to design approximation algorithms for several problems, including vertex cover problem, set cover problem and feedback vertex set problem.

2 Vertex Cover

Given an undirected graph $G = (V, E)$, a subset of vertices $U \subseteq V$ is called a vertex cover if for each edge in E, at least one of its adjacent vertex is in U. Finding a minimum vertex cover is called vertex cover problem. Now, let's first relax this problem as a linear programming problem:

$$\min \sum_{v \in V} x_v \tag{1}$$
$$s.t.\ \forall e = (u, v) \in E \quad x_u + x_v \geq 1 \tag{2}$$
$$\forall v \in V \qquad x_v \geq 0 \tag{3}$$

We call the above linear programming problem the Primal problem. Let's write down its Dual form, where E_v denotes all the edges adjacent to vertex v:

$$\max \sum_{e \in E} y_e \tag{4}$$
$$s.t.\ \forall v \in V \quad \textstyle\sum_{e:e \in E_v} y_e \leq 1 \tag{5}$$
$$\forall e \in E \qquad y_e \geq 0 \tag{6}$$

Before introducing the primal-dual algorithm for vertex cover problem. Let's first give some general ideas for primal-dual method.

As Figure 1 shows, due to weak duality, the cost of any primal solution is larger than or equal to the cost of any dual solution. In any Primal-Dual algorithm, we will construct an integral primal solution (Algo. (P)) and a (fractional) dual solution (Algo. (D)), and bound the approximation ratio (Approx. Ratio) by the ratio (Algo. Ratio) between the cost of primal solution and the cost of dual solution. We note that in a Primal-Dual algorithm, we don't need to solve the primal linear programming. This is a central difference from the Rounding method.

Now let's see the Primal-Dual algorithm for vertex cover problem. Initially, we set all $x_v,\ v \in V$ and all $y_e,\ e \in E$ to zero. Then, we choose an arbitrary dual variable y_e. We increase y_e until one of the related dual constraint becomes tight, which means this constraint becomes a equality. Suppose the primal variable corresponding to this tight dual constraint is x_v, we set x_v to 1. Repeat this process until all primal constraints are satisfied. In other words, we start from an infeasible primal solution and a feasible but far

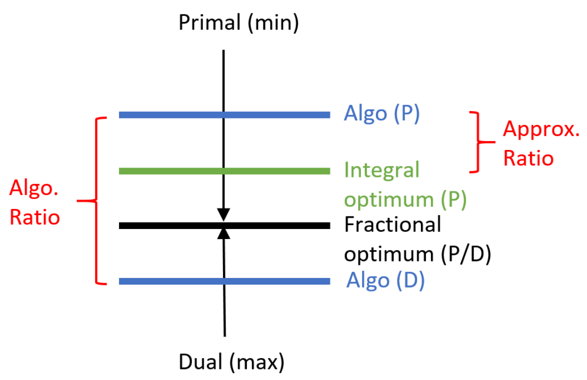

Figure 1: Primal-Dual Method.

from optimal dual solution, then we gradually increase primal variables and dual variables to make primal solution feasible and dual solution near optimal. Obviously, this algorithm outputs a vertex cover. Now we claim this algorithm achieves 2-approximation.

Theorem 1. *There is an algorithm achieving 2-approximation for minimizing vertex cover.*

Proof. We can just bound the ratio between the cost of primal solution and the cost of dual solution. Denote the cost of primal solution by P, and the cost of dual solution by D.

$$\begin{aligned} P &= \sum_{v\in V:x_v=1} x_v &(7)\\ &= \sum_{v\in V:x_v=1}\sum_{e\in E_v} y_e &(8)\\ &\leq 2\sum_{e\in E} y_e &(9)\\ &= 2D &(10)\end{aligned}$$

The second equality is because that a primal variable is set to 1 only if its corresponding dual constraint is tight. The only inequality is due to the fact that each edge has only two adjacent vertices, which means each dual variable at most appears twice in the sum. □

3 Set Cover

Next, let's see a more general problem, set cover problem. Given a set of elements $U = \{1, 2, \ldots, n\}$ (called the universe) and a collection $\mathbb{C}$ of m sets whose union equals the universe, the set cover problem is to find the smallest sub-collection of $\mathbb{C}$ whose union equals the universe. Just as before, let's first write down the primal and dual formulation.

Primal:

$$\begin{aligned} &\min \sum_{S\in\mathbb{C}} x_S &(11)\\ s.t.\ \forall e\in U \quad &\textstyle\sum_{S:e\in S} x_S \geq 1 &(12)\\ \forall S\in\mathbb{C} \quad &x_S \geq 0 &(13)\end{aligned}$$

Dual:

$$\max \sum_{e \in U} y_e \tag{14}$$
$$s.t.\ \forall S \in \mathbb{C} \quad \textstyle\sum_{e \in S} y_e \leq 1 \tag{15}$$
$$\forall e \in U \qquad y_e \geq 0 \tag{16}$$

The Primal-Dual algorithm for set cover problem is exactly the same one for vertex cover. We claim that this algorithm can achieve f-approximation ratio for set cover problem, where f is the maximum number of sets that an element belongs to.

Theorem 2. *There is an algorithm achieving f-approximation for minimizing set cover, where f is the maximum number of sets that an element belongs to.*

Proof. Just as before, we can just bound the ratio between the cost of primal solution and the cost of dual solution. Denote the cost of primal solution by P, and the cost of dual solution by D.

$$\begin{aligned} P &= \sum_{S \in \mathbb{C}: x_S = 1} x_S && (17)\\ &= \sum_{S \in \mathbb{C}: x_S = 1} \sum_{e \in S} y_e && (18)\\ &\leq f \sum_{e \in U} y_e && (19)\\ &= f \times D && (20) \end{aligned}$$

The second equality is because a primal variable is set to 1 only if its corresponding dual constraint is tight. The only inequality is due to the fact that each element belongs to at most f sets, which means each dual variable appears at most f times in the sum. □

4 Feedback Vertex Set

Next, let's move on to a more interesting problem, feedback vertex set problem. Given an undirected graph $G = (V, E)$, the feedback vertex set problem is to find the smallest subset of vertices, whose removal makes the graph acyclic. An immediate observation is that this is just a special case of set cover problem, if we view cycles as 'elements' and vertices as 'sets'. Let's first write down the primal and dual formulation of feedback vertex set problem.

$$\min \sum_{v \in V} x_v \tag{21}$$
$$s.t.\ \forall \text{cycle } C \quad \textstyle\sum_{v \in C} x_v \geq 1 \tag{22}$$
$$\forall v \in V \qquad x_v \geq 0 \tag{23}$$

Dual:

$$\max \sum_{\text{cycle } C} y_C \tag{24}$$

$$s.t.\ \forall v \in V \quad \textstyle\sum_{C:v\in C} y_C \le 1 \tag{25}$$

$$\forall \text{cycle } C \qquad y_C \ge 0 \tag{26}$$

Since feedback vertex set problem is a special case of set cover problem, a natural question is can we directly use the previous algorithm to tackle feedback vertex set problem? Let's see what's the value of f in feedback vertex set problem. We find f equals the maximum number of vertices on a cycle, which could be n if the graph is just a giant cycle. An n-approximation algorithm is totally useless, since even simply removing all vertices yields an n-approximation algorithm (assume at least one cycle exists). It seems like we need to modify the algorithm and give a more careful analysis.

Let's investigate the relationship between primal solution and dual solution in feedback vertex problem (suppose the feedback vertex set chosen is F):

$$\begin{aligned} P &= \sum_{v\in F} x_v & (27)\\ &= \sum_{v\in F}\sum_{C:v\in C} y_C & (28)\\ &= \sum_{\text{cycle } C} |F\cap C| y_C & (29)\end{aligned}$$

One observation is that if all $|F \cap C|$ for nonzero y_C is small, we can gain a much better approximation ratio. It seems like we'd better only increase dual variables whose corresponding cycles are short. However, what if the graph is just a giant cycle? Actually, in this case, just removing one vertex is enough. In order to exploit this observation, we add a 'degree-1 removal' step into the original algorithm. In each step, before we choose a dual variable to increase, we first remove all degree-1 vertices of the graph until there is no degree-1 vertices. Another observation is that degree-2 vertex is not a big trouble. Given a long degree-2 path, we can just choose one of them, and the 'degree-1 removal' step will remove the remainings. What we should really worry about is vertices with high degree (≥ 3). Thus, instead of choosing an arbitrary dual variable to increase, now we choose the dual variable whose corresponding cycle contains smallest number of high degree (≥ 3) vertices. This selection procedure can be solved using a BFS-like algorithm within polynomial time.

We first prove a crucial lemma claiming that on a degree-1 free graph, there must exist a cycle whose high degree (≥ 3) vertices are no more than $2\log n$, where n is the number of vertices on this graph.

Lemma 3. *Given a graph with no degree-1 vertex, it must contain a cycle with no more than* $2\log n$ *high degree (*≥ 3*) vertices, where n is the total number of vertices on this graph.*

Proof. In this proof, we define high degree vertices as vertices with degree larger than or equal to 3. First, we shortcut all degree-2 vertices on the graph. If there is a path containing only degree-2 vertices between two high degree vertices, we simply substitute this path by an edge between these two high degree vertices.

After shortcuting all degree-2 paths, we have a graph containing only high degree vertices. The number of high degree vertices is no more than n. Starting from any vertex, we do a Breadth-First Search of this graph. On the search tree, each vertex has at least two children. Thus, the depth of the search tree cannot be larger than $\log n$. There must exist an edge crossing different paths of this search tree, which immediately constitutes a cycle with no more than $2\log n$ high degree vertices. ☐

With the above lemma, we know the cycle we choose in the algorithm has no more than $2\log n$ high degree vertices. Between these high degree vertices, there could be no more than $2\log n$ degree-2 segments. We claim for each degree-2 segment, at most one vertex can be put into the feedback vertex set. This is because once we put one vertex from the degree-2 segment into the feedback set, the ‘degree-1 removal’ procedure will remove all the remaining vertices on this degree-2 segments. It turns out that on each chosen cycle, at most $4\log n$ vertices in total can be put into the feedback vertex set. In other words, we have proven that for any nonzero dual variable y_C, $|F \cap C|$ is upper bounded by $4\log n$, which immediately implies that our algorithm achieves $4\log n$ approximation ratio.

Theorem 4. *There is an algorithm achieving $4\log n$-approximation ratio for minimizing feedback vertex set.*

The major lesson from this algorithm is that sometimes we should be careful on choosing which dual variable to increase, instead of just arbitrarily choosing any dual variable.

5 Summary

In this lecture, we have given the general framework of Primal-Dual method. We have showed how to design and analyze Primal-Dual algorithms through three examples, vertex cover problem, set cover problem and feedback vertex set problem. Through the analysis of feedback vertex set problem, we have learned that sometimes we should be careful on choosing which dual variable to increase.