Analysis of Algorithms	**LP-Based Approximation**
CS6820 Fall 2022	**Monday, November 28, 2022**

1 Linear Programming

Linear programming (LP) is the problem of optimizing (minimizing or maximizing) a linear objective function subject to linear constraints. A *linear program* with m constraints and n variables is defined by vectors $b \in \mathbb{R}^m$ and $c \in \mathbb{R}^n$, a vector x of n variables ranging over $\mathbb{R}$, and an $m \times n$ matrix $A \in \mathbb{R}^{m\times n}$. The LP is

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \geq b \text{ and } x \geq 0. \end{array}$$

Here we are thinking of the vectors as column vectors, and the superscript T refers to the transpose (making a row vector out of a column vector). The expressions $c^T x$ and Ax refer to ordinary inner product and matrix-vector multiplication, respectively.

This is the standard form of an LP minimization problem. One could formulate a more general version without the constraints $x \geq 0$, but any such system can be transformed to the standard form without loss of generality.

The constraints define a closed polyhedron in $\mathbb{R}^n$. This polyhedron is called the *feasible region* of the LP, and the LP is called *feasible* if the feasible region is nonempty; that is, if constraints have any solution at all.

Let us call this the *primal* LP. Associated with the primal LP is a *dual* LP involving a vector y of m real variables:

$$\begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to} & A^T y \leq c \text{ and } y \geq 0 \end{array}$$

or equivalently,

$$\begin{array}{ll} \text{maximize} & y^T b \\ \text{subject to} & y^T A \leq c^T \text{ and } y \geq 0. \end{array}$$

The feasible region of the dual LP is a polyhedron in $\mathbb{R}^m$. Note that the dual is a maximization problem and the inequalities $A^T y \leq c$ go in the other direction. The dual of the dual is the original primal.

Lemma 1 (Weak Duality)**.** *If x and y are points in the feasible regions of the primal and dual LPs, respectively, then $y^T b \leq c^T x$; that is, the value of the dual objective function at any feasible point y is a lower bound on the value of the primal objective function at any feasible point x.*

Proof. Let x and y be any points in the feasible regions of the primal and dual LPs, respectively. Since $Ax \geq b$ and $y \geq 0$, we have $y^T Ax \geq y^T b$. Since $y^T A \leq c^T$ and $x \geq 0$, we have

$y^T Ax \leq c^T x$. Putting these together, $y^T b \leq y^T Ax \leq c^T x$. □

It follows from weak duality that if the primal LP is feasible, then it has an optimal solution if the dual LP is feasible, since the values of the primal objective function are bounded below and the feasible region of the primal LP is closed. Similarly, if the dual LP is feasible, then it has an optimal solution if the primal LP is feasible.

Lemma 2 (Strong Duality)**.** *If both the primal and dual LPs are feasible, then their optimal values are equal.*

We will skip the proof of this lemma, as it is quite difficult. Some slides with a proof outline can be found here. The max flow-min cut theorem of network flow theory is an example of linear programming duality.

If we require integer solutions to a given linear program, the problem is called *integer programming* and is NP-complete. However, there are polynomial-time algorithms for linear programming without this restriction, the *ellipsoid algorithm* due to Khatchian [3] and the *interior point method* due to Karmarkar [2]. The much older *simplex method*, due to Dantzig [1], is exponential in the worst case but tends to work well in practice.

2 LP-Based Approximation Algorithms

Linear programming is an extremely versatile technique for designing approximation algorithms for NP-complete problems, because it is one of the most general and expressive problems that we know how to solve in polynomial time. In the next few lectures we will discuss some applications of linear programming to the design and analysis of approximation algorithms.

2.1 LP Rounding Algorithm for Weighted Vertex Cover

In an undirected graph $G = (V, E)$, if $S \subseteq V$ is a set of vertices and e is an edge, we say that S *covers* e if at least one endpoint of e belongs to S. We say that S is a *vertex cover* if it covers every edge. In the weighted vertex cover problem, one is given an undirected graph $G = (V, E)$ and a weight function $w : V \to \mathbb{R}_+$, and one must find a vertex cover of minimum total weight.

We can express the weighted vertex cover problem as an integer program by using *decision variables* x_v for all $v \in V$ that encode whether $v \in S$. For any set $S \subseteq V$ we can define a vector x, with components indexed by vertices of G, by specifying that

$$x_v = \begin{cases} 1, & \text{if } v \in S, \\ 0, & \text{otherwise.} \end{cases}$$

The set S is a vertex cover if and only if the constraint $x_u + x_v \geq 1$ is satisfied for every edge $(u, v) \in E$. Conversely, if $x \in \{0,1\}^V$ satisfies $x_u + x_v \geq 1$ for every $(u, v) \in E$, then $S = \{v \mid x_v = 1\}$ is a vertex cover. Thus, the weighted vertex cover problem can be expressed as the following integer program.

$$\begin{array}{lll} \text{minimize} & \sum_{v \in V} w_v x_v & \\ \text{subject to} & x_u + x_v \geq 1 & \text{for } e = (u,v) \in E \\ & x_v \in \{0,1\} & \text{for } v \in V \end{array} \tag{1}$$

To design an approximation algorithm for weighted vertex cover, we will transform this integer program into a linear program by relaxing the constraint $x_v \in \{0,1\}$ to allow the variables x_v to take fractional values.

$$\begin{array}{lll} \text{minimize} & \sum_{v \in V} w_v x_v & \\ \text{subject to} & x_u + x_v \geq 1 & \text{for } e = (u,v) \in E \\ & x_v \geq 0 & \text{for } v \in V \end{array} \tag{2}$$

It may seem more natural to replace the constraint $x_v \in \{0,1\}$ with $x_v \in [0,1]$ rather than $x_v \geq 0$, but it does not matter; an optimal solution of the linear program will never assign any of the variables x_v a value strictly greater than 1, as the value of any such variable could always be reduced to 1 without violating any constraints and would only improve the objective function $\sum_v w_v x_v$. Thus, writing the constraint as $x_v \geq 0$ rather than $x_v \in [0,1]$ is without loss of generality.

It is possible that a fractional solution of (2) can achieve a strictly lower weight than any integer solution. For example, let G be a 3-cycle with vertices u, v, w, each having weight 1. Then the vector $x = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$ satisfies all of the constraints of (2) and the objective function evaluates to $\frac{3}{2}$ at x; in fact, this is the optimal solution. In contrast, the minimum weight of a vertex cover is 2.

We can solve the linear program (2) in polynomial time, but as we have just seen, the optimal solution may be fractional. In that case, we need to figure out how to post-process the fractional solution to obtain an actual vertex cover. In this case, the natural idea of rounding to the nearest integer works. Let x be an optimal solution of the linear program (2) and define

$$\widetilde{x}_v = \begin{cases} 1, & \text{if } x_v \geq \frac{1}{2}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Let $S = \{v \mid \widetilde{x}_v = 1\}$. Then S is a vertex cover, because for every edge (u, v), the constraint $x_u + x_v \geq 1$ implies that at least one of x_u, x_v is at least $1/2$, so at least one of $\widetilde{x}_u, \widetilde{x}_v$ is 1.

Finally, to analyze the approximation ratio, we observe that the rounding rule (3) has the property that for all v,

$$\widetilde{x}_v \leq 2x_v.$$

Letting S denote the vertex cover chosen by our LP rounding algorithm, and letting OPT denote the optimum vertex cover, we have

$$\sum_{v \in S} w_v \;=\; \sum_{v \in V} w_v \widetilde{x}_v \;\leq\; 2 \sum_{v \in V} w_v x_v \;\leq\; 2 \sum_{v \in \mathrm{OPT}} w_v,$$

where the final inequality holds because the fractional optimum of the linear program (2) must be less than or equal to the optimum of the integer program (1) because its feasible region is at least as big.

2.2 Primal-Dual Algorithm for Weighted Vertex Cover

The algorithm presented in the preceding section runs in polynomial time and outputs a vertex cover whose weight is at most twice the optimal weight, a fact that we express by saying that its *approximation factor* (or *approximation ratio*) is 2.

However, the algorithm needs to solve a linear program. Although this can be done in polynomial time, there are much faster ways to compute a vertex cover with approximation factor 2 without solving a linear program. One such algorithm, which we present in this section, is a *primal-dual approximation algorithm*, meaning that it makes choices guided by the linear program (2) and its dual but does not actually solve them to optimality.

Let us write the linear programming relaxation of weighted vertex cover once again, along with its dual.

$$\begin{array}{lll} \text{minimize} & \sum_{v \in V} w_v x_v & \\ \text{subject to} & x_u + x_v \geq 1 & \text{for } e = (u,v) \in E \\ & x_v \geq 0 & \text{for } v \in V \end{array} \tag{4}$$

$$\begin{array}{lll} \text{maximize} & \sum_{e \in E} y_e & \\ \text{subject to} & \sum_{e \in \delta(v)} y_e \leq w_v & \text{for } v \in V \\ & y_e \geq 0 & \text{for } e = E \end{array} \tag{5}$$

Here, the notation $\delta(v)$ denotes the set of all edges having v as an endpoint. One can interpret the dual LP variable y_e as *prices* associated to the edges, and one can interpret w_v as the *wealth* of vertex v. The dual constraint $\sum_{e \in \delta(v)} y_e \leq w_v$ asserts that v has enough wealth to pay for all of the edges incident to it. If edge prices satisfy all the constraints of (5) then every vertex has enough wealth to pay for its incident edges, and consequently every vertex set S has enough combined wealth to pay for all of the edges covered by S. In particular, if S is a vertex cover, then the combined wealth of the vertices in S must be at least $\sum_{e \in E} y_e$, which is a manifestation of *weak duality*: if $\sum_{v \in V} w_v x_v$ is the optimum calue of the primal LP, then

$$\sum_{e \in E} y_e \;\leq\; \sum_{v \in V} w_v x_v \;\leq\; \sum_{v \in \mathrm{OPT}} w_v,$$

that is, the optimum value of the dual LP is a lower bound on the optimum value of the primal LP, which is a lower bound on the weighted vertex cover problem.

The dual LP insists that we maximize the combined price of all edges, subject to the constraint that each vertex has enough wealth to pay for all the edges it covers. Rather than maximizing the combined price of all edges exactly, we will set edge prices using a natural (but suboptimal) greedy heuristic: go through the edges in arbitrary order, setting the price of each one as high as possible without violating the dual constraints. This results in the following algorithm.

Algorithm 1 Primal-dual algorithm for vertex cover

Initialize $S \leftarrow \varnothing$ and $s_v \leftarrow 0$ for all $v \in V$
for all $e = (u, v) \in E$ **do**
    $y_e \leftarrow \min\{w_u - s_u, w_v - s_v\}$
    $s_u \leftarrow s_u + y_e$
    $s_v \leftarrow s_v + y_e$
    if $s_u = w_u$ **then** $S \leftarrow S \cup \{u\}$
    else $S \leftarrow S \cup \{v\}$
    end if
end for
return S

The variables s_v keep track of the sum $\sum_{e \in \delta(v)} y_e$ (i.e., the left-hand side of the dual constraint corresponding to vertex v) as it grows during the execution of the algorithm.

It is clear that each iteration of the main loop runs in constant time, so the algorithm runs in linear time. After processing the edge (u, v), at least one of the vertices u, v must belong to S, so S is a vertex cover.

To conclude the analysis, we need to prove that the approximation factor is 2. We note the following loop invariants hold initially and are preserved by the body of the **for** loop, therefore are also true at the end:

1. y is a feasible vector for the dual linear program;
2. $s_v = \sum_{e \in \delta(v)} y_e$;
3. $S \subseteq \{v \mid s_v = w_v\}$;
4. $\sum_{v \in V} s_v = 2 \sum_{e \in E} y_e$.

Now the proof of the approximation factor is easy. Recalling that $\sum_{e \in E} y_e \leq \sum_{v \in \mathrm{OPT}} w_v$ by weak duality, we find that

$$\sum_{v \in S} w_v \; = \; \sum_{v \in S} s_v \; \leq \; \sum_{v \in V} s_v \; = \; 2 \sum_{e \in E} y_e \; \leq \; 2 \sum_{v \in \mathrm{OPT}} w_v.$$

In Algorithm 1, the rule for updating S by inserting each vertex v such that $s_v = w_v$ is inspired by the principle of *complementary slackness* from the theory of LP duality. If x^* and y^* are optimal solutions of the primal and dual LPs respectively, then for every i such that $x_i^* > 0$, the i^{th} dual constraint is satisfied with equality by y^*. Similarly, for every j such that $y_j^* > 0$, the j^{th} primal constraint is satisfied with equality by x^*. Thus, it is natural that our decisions of which vertices to include in our vertex cover in the primal solution should be guided by which dual constraints are tight ($s_v = w_v$).

References

[1] G. B. Dantzig. Linear programming. In *Proc. Symp. Modern Calculating Machinery and Numerical Methods*, pages 29–31, July 1948.

[2] Narendra Karmarkar. A new polynomial time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[3] L. G. Khatchian. A polynomial algorithm in linear programming. *Doklady Akademii Nauk SSSR*, 244:1093–1096, 1979.