

Lecture 8: Linear Programs

Maximum flow and many other applications can be incorporated into a broader class of problems called Linear Programs (LP). There are several courses at Stanford that deal specifically with LP (such as MS&E 310). The algorithm for solving LP known as the *simplex method* has been called one of the most important algorithms of the 20th century.

1. The General, the Cook and the Mother of all max-min theorems

Suppose we have a division of soldiers, a General and a Cook. There is a set of food ingredients with desirable property j (such as calories, protein, vitamins, etc...) and we want to ensure that each soldier gets a sufficient amount of each property. Each available food type i (like potatoes, chicken, tofu...) has a certain quantity a_{ij} of property j . Each unit of food type i costs c_i , and soldier has to receive at least r_j quantity of property j (for instance 2000 kCal). The General wants the cook to prepare a meal (regardless of taste) that meets all of the soldiers requirements at a minimum cost. This can be formally written as follows using variables x_i to indicate the amount of food of type i used.

$$\begin{aligned} \text{minimize: } & \sum_i c_i x_i \\ \text{subject to: } & \sum_i a_{ij} x_i \geq r_j \quad \text{for all } j \\ & x_i \geq 0 \quad \text{for all } i \end{aligned}$$

Such problems are called LP because we optimize over a linear objective function with linear constraints.

Assume we want to minimize $4x_1 + 3x_2$ subject to $x_1 + x_2 \geq 10$ and $2x_1 + x_2 \geq 15$, as usual $x_1, x_2 \geq 0$. For instance potatoes cost 3 dollars a pound and steak 4 dollars the ounce. Assume both foods have the same amount of protein per unit of weight, and that steak has half the carbs of potatoes. The General wants the food to be as cheap as possible, but each soldier must get at least 10 units of protein and 15 of carbs.

One solution to such problem is $x_1 = 5$, $x_2 = 5$, which yields a cost of 35. But how can we prove it is optimal? This time is quite simple because we can see that if we add 2 times the first constraint to the second, we get $4x_1 + 3x_2 \geq 35$, hence any feasible solution has to be at least 35.

We could have tried another approach, namely for each constraint, assign a variable (here α_1 and α_2) and add a constraint for each of the variables. We get the dual problem: maximize

$10\alpha_1 + 15\alpha_2$ subject to $\alpha_1 + 2\alpha_2 \leq 4$ and $\alpha_1 + \alpha_2 \leq 3$ (of course $\alpha_1, \alpha_2 \geq 0$). One can prove that any feasible solution to the dual yields a lower bound on the value of the primal (and vice-versa). That is called weak duality.

One can easily see here that the value of the optimum for the primal and the dual are the same here. That is true in general, and is called strong duality.

Definition: Linear Program

Let $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$. The following is the general form of an LP:

$$\min_{x \in \mathbb{R}^n} (c^T x) \text{ subject to } Ax \geq b \text{ and } x \geq 0$$

or equivalently,

$$\max_{y \in \mathbb{R}^m} (b^T y) \text{ subject to } A^T y \leq c \text{ and } y \geq 0$$

One is called the primal, the other is the dual.

Theorem 1 *Weak and Strong Duality Theorems*

Let $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, A $m \times n$ matrix. Consider the following LP:

$$\min_{x \in \mathbb{R}^n} (c^T x) \text{ subject to } Ax \geq b \text{ and } x \geq 0$$

and its dual

$$\max_{y \in \mathbb{R}^m} (b^T y) \text{ subject to } A^T y \leq c \text{ and } y \geq 0$$

Then

- *Weak duality:* $\forall x \in \mathbb{R}^n$ feasible for the primal, and $\forall y \in \mathbb{R}^m$ feasible for the dual, we have that $c^T x \geq b^T y$
- *Strong Duality:* if x^* and y^* are the optimal solutions of the primal and dual LP then $c^T x^* = b^T y^*$

There is some debate as to who first proved the Strong Duality theorem, but the first historical reference goes back to a conversation between Von Neumann and Dantzig.

This can be seen as a generalization to the Min-cut/Max-flow theorem, as one can formulate maximum flow as a LP. The dual LP then corresponds to the minimum cut problem. There exist polynomial time algorithms to solve LP (such as the ellipsoid method, interior point methods) but no strongly polynomial algorithms are known (recall these are algorithms whose running time is only dependent on the number of variables and the number of constraints).

Note that for the combinatorial problems that we presented before (e.g. maximum matching, maximum flow, min-cost flow etc...) strongly polynomial time algorithms are known. In fact, almost all the algorithms that we presented in the class are strongly polynomial.

2. The Simplex Algorithm

The first and most famous algorithm for solving linear programs was introduced by George Dantzig in 1947. At its core is the realization that the feasible region of a linear program represents a closed convex polytope (also known as a *simplex*) with a linear objective function. Therefore, it is easy to see that an optimal solution must lie on one of the vertices of the polytope.

We can find a vertex of the polytope by taking a set of n linearly independent constraints and performing gaussian elimination to find their intersection. If we just check each possible combination, we will eventually reach the optimal solution. The problem, of course, is that there are $\binom{m}{n}$ vertices, where n is the number of variables and m is the number of constraint, which is in general exponential in n and m .

Dantzig's idea was to walk along the vertices of the polytope, stepping only to neighboring vertices that had a better objective value. This is basically just the "local search" algorithm for a walk in a polytope. How neighboring vertices are picked is known as the "pivot rule". Does the simplex always return an optimal value? Yes, because it is finding the minimum of a convex region, and in a convex region any local minimum is also a global minimum.

The simplex algorithm was widely successful and is still in broad use today. But does it have polynomial running time? After 60 years of research this question is still open. Most commonly used pivot rules are shown to take an exponential number of steps for certain "degenerate" cases. For most applications, however, simplex works very well.

3. Integer solutions

Example: Min-cut/max-flow problem

We can write the min-cut/max-flow problem as a linear program in the variables f_{ij} , the flow on edge $(i, j) \in E$ with capacity $cap(i, j)$:

$$\begin{aligned}
 \text{maximize:} \quad & \sum_{(s,j) \in E} f_{sj} \\
 \text{subject to:} \quad & \sum_{(i,j) \in E} f_{ij} - \sum_{(j,i) \in E} f_{ji} = 0 \quad \text{for all } i \notin \{s, t\} \\
 & f_{ij} \leq cap(i, j) \\
 & f_{ij} \geq 0
 \end{aligned}$$

In this formulation we are maximizing the flow out of the source s and imposing constraints of flow conservation and capacity bounds. Note that to write this in standard form one can express an equality constraint as a pair of \leq and \geq constraints.

In general, how much harder can LP be when we add the extra requirement that the solution must be an integer? While this may seem easier, since there are only an exponential number of possible solutions instead of infinitely many, it is generally a lot harder. Such problems are called integer programs, and a lot of known NP-hard problems can be formulated as integer programs.

Even if integer programs are usually very hard, sometimes one “gets lucky” and we can solve it as an LP and the LP itself yields an integer solution.

Definition: Totally Unimodular Matrices

A matrix A is said to be totally unimodular if all its square submatrices have determinants in the set $\{-1, 0, 1\}$.

All entries in a totally unimodular matrix are in $\{-1, 0, 1\}$. This is obvious given that entries are simply submatrices of size 1.

Theorem 2 (Hoffman and Kruskal, 1956) *A characterization of LPs with integral solutions:*

Consider the LP, $\max(c^T x)$ subject to $Ax \leq b$ and $x \geq 0$ and its dual. The primal and dual linear programs have an integral optimum (if they are finite) solution for all integral vectors b and c if and only if an integral matrix A is totally unimodular.

Proof: “ \Rightarrow ” Let A be totally unimodular. The solution x is a vertex of the polyhedron defined by the constraints of the LP,

$$\begin{pmatrix} A \\ -I \end{pmatrix} x \leq \begin{pmatrix} b \\ 0 \end{pmatrix} \quad (1)$$

and satisfies an $n \times n$ nonsingular subsystem $A'x = b'$ of (1). Since the matrix A is totally unimodular, $|\det(A')| = 1$, and by Cramer’s rule $x = (A')^{-1}b'$ is integral for integral b .

“ \Leftarrow ” Suppose that the vertices of the polygon are integral for integral A and b . Let A be an $m \times n$ matrix with some number $k \leq m$ of linearly independent columns arranged as the first k columns of this matrix. Consider the system

$$\begin{pmatrix} A & I_m \end{pmatrix} z' = \begin{pmatrix} A & I_m \end{pmatrix} \begin{pmatrix} z'' \\ b - Az'' \end{pmatrix} = b \quad (2)$$

where I_m is a $m \times m$ identity matrix. The vector z' is composed of a n vector $z'' = \begin{pmatrix} w \\ 0 \end{pmatrix}$ and m slack variables. The subsystem $A'w = b'$ has a unique solution for a submatrix A'

consisting of the first k rows and columns of A and with b' consisting of the first k entries of b . We now have constructed a vertex z'' of the polyhedron by satisfying n of the following inequalities exactly,

$$\begin{pmatrix} A \\ I_n \end{pmatrix} z'' \leq \begin{pmatrix} b' \\ 0 \end{pmatrix}$$

namely the first k and the last $n - k$. By our assumption this vertex is integral.

We now construct an $m \times m$ nonsingular matrix B from system (2) by selecting the first k (linearly independent) and the last $m - k$ columns. We now have $|\det(B)| = |\det(A')|$. Furthermore, we can express b as follows. For $i \in \{1, \dots, m\}$ pick an integral y such that $b = By + e_i = Bz$ where z corresponds to the first k and the last $m - k$ entries of z' .

Since z'' is integral we have that z' is integral (for integral A and b). Then z is integral and by construction so is $B^{-1}e_i$. Since i is arbitrary we conclude that B^{-1} is integral. Since $1 = \det(B)\det(B^{-1})$ we have $|\det(A')| = |\det(B^{-1})| = 1$. Thus all submatrices of A have determinant in $\{-1, 0, 1\}$ and A is totally unimodular as required. ■

The above proof is due to Veinott and Dantzig [1968].