CS270 Algorithms & Data Structures **Spring 2008**

Lecture 10 linear programming : 2.19.08

Lecturer: Satish Rao *Scribe: Rao, for now*

Disclaimer: *These are rough notes, with some exercises.*

10.1 From last time...

Question: We'll work with standard form. What is some intuition here?

Well, we have an $m \times n$ matrix A (where $n > m$) and constraints $x_i \geq 0$. A vertex has $Ax = b$ and a set of $n - m$ variables $x_i = 0$. (We will call these the non-basic variables.)

As suggested in the book, we start at such a solution. We consider the subset of m columns in the matrix A that correspond to the (possibly) nonzero variables (basic variables). We refer to it as A_B.

Question: Can we easily determine the value of these basic variables?

Sure $A[x_B, x_N] = b$ or $A_B x_B = b$ or $x_B = A_B^{-1} b$.

Question: When does a basis (subset of m columns) correspond to a feasible solution?

When all the values are positive.

Question: Let's assume we have a basic feasible solution. How would we proceed?

We write the linear program as minimizing $c_B x_B + c_N x_N$ where

$$A_B x_B + A_N x_N = b,$$

where $x_B, x_N \geq 0$.

Question: For any point in the feasible region, if we know the value of cordinates x_N, what is the value of the coordinates x_B?

Well, $A[x_B, x_N] = b$, or $Ax_B = b - A_N x_N$, and

$$x_B = A_B^{-1} b - A_B^{-1} A_N x_N.$$

That is, the x_N coordinates provide a coordinate system for any feasible point.

Question: What is the cost in terms of just x_N?

$$\begin{aligned} c[x_B, x_N] &= c_B x_B + c_N x_N & (10.1) \\ &= c_B(A_B^{-1} b - A_B^{-1} A_N x_N) + c_N x_N & (10.2) \\ &= c_B A_B^{-1} b + (c_N - c_B A_B^{-1} A_N) x_N & (10.3) \\ & & (10.4) \end{aligned}$$

The current cost is the first term (since x_N's are zero). We now have a new cost function $\tilde{c} = c_N - c_B A_B^{-1} A_N$ on the non-basic variables.

Question: How to improve?

Well, make some x_N nonzero. Which one, one that improves the cost, i.e., one where $\tilde{c}$ is negative, say x_l.

Question: When we increase some non-basic x_l, what happens to the basic variables?

They change. So, we only increase as long as $x_B = A_B^{-1}b - A_B^{-1}A_N x_N$ remains positive on all coordinates. Thus, the index l where the

$$A_B^{-1}b/A_B^{-1}A_N,$$

(where dividiing is coordinate-wise) is minimal. We set this coordinate to zero, and allow l to be non-zero, and can rest assured that the other x_B's stay positive.

Then, we add the column A_l to the basis, and remove A_i from the basis.

Question: How much time does such a step take?

$O(n^3)$ for the inversion and $O(nm)$ for determining a variable.

Or with something called rank 1 updates, one can recompute A_B^{-1} in $O(nm)$ time. So, the total time becomes $O(nm)$ per iteration.

10.2 The tableau.

Question: Ugh, perhaps let's be more specific. Do you recall a maximization version?

We'll consider a maximization problem

$$A_0 x \leq b, x \geq 0,$$

where we have are maximizing $c_0 x$.

Question: Convert it to $Ax = b$ by adding m slack variables to produce a matrix $A_0|I$, that is A_0 plus a set of columns that corresponds to the identity matrix. How should one organize it?

A tableau, on the board (I should write it out) but for in matrix form.

$$\begin{array}{rcl} A_0| & I & |b \\ -c_0| & 0 & |0 \end{array}$$

We can associate this matrix with the simplex notions above; the basic variables are the slack variables $(x_n, \ldots, x_{n+m})$, the zero ones are the old variables. A_B is the identity (and thus, A_B^-1 is easy to compute.

Question: What is the solution?

The slack variables $x_{n+i} = b_i$.. The right column, assigned to the basic variables.

Question: What is the right column?

A_B^-1b. Well A_B is identity!

Question: What is the value of the solution?

0. Lower left hand corner. c_B is 0, and only these variables are non-zero.

Question: What is the bottom row?

The nonzeros form

$$(c - c_B A_B^{-1} A).$$

Note, this is zero for indices in the basis since A_B^{-1} times a basis column give the indicator vector for the relevant row.

Question: Hey, is $A_B^{-1} A_N$ somwhere?

Sure, that happens to be A_0.

Question: How to take a step?

Choose a column where c is the most negative. Choose the row where $b_i/a_{i,e}$ is minimal. Pivot; do row operations until only this one is nonzero.

Question: What is A_B now?

The matrix corresponding to the identity?

Question: Where is A_B^{-1}?

The matrix corresponding to the slack variables. Why?

Question: Recall, gaussian elmination to produce an inverse?

Take $A|I$ and do row operations to make A the identity. Then the last n columns is A^{-1}. Basically consider first "zeroing out" the first column of I basically encodes the row operations needed to zero out A. Then, induction..

Question: What is to verify?

Last column is $A_B^{-1}b$, the submatrix consisting of nonbasic columns is $A_B^{-1}A_N$, bottom row is $\tilde{c} = c - c_B A_B^{-1} A_N$. The first two from the notion that multiplying by the inverse amounts to performing the row operations required to make A the identity, i.e., the fact that Gaussian elimination makes the dual. The third follows similarly, (inductively is perhaps easier to think about.)

Question: Is the value ok?

Sure, we basically add the right amount to the $\tilde{c_j} A_B^{-1} b_i / A_{ij}$ to the lower right value.

Exercise 1: Setup and run the ruby lp tableau program under the lecture on your computer, or write one yourself.

Exercise 1.1: Describe how to recover the solution from the final tableau. Exercise 1.2: Recover the one for the linear program presented. Exercise 1.3: What problem does the linear program solve? Exercise 1.4: Recover the the dual solution. Exercise 1.5: What does it correspond to?

10.3 Duality.

Question: Did you read about duality?

Yes. We consider the canonical form of linear programs as

$$\max cx$$
$$Ax \leq b$$
$$x \geq 0.$$

The dual is

$$\min by$$
$$y^T A \geq c$$
$$y \geq 0.$$

Ignoring, for now unconstrained variables, equalities, etc.

Question: What?

You can think of the dual as having a variable for each equation, and an equation for each variable. For example, for maximum flow, we introduce a variable for each conservation constraint, d_i, and a variable for each capacity constraint with right hand side the coefficient of the variable in the optimization function. (We add a equation in the primal that consists of the flow out of the source and have a variable for that as well, adding a variable f to maximize.)

That is $\max f$ subject to

$$\forall e, f_e \leq c_e.$$
$$\sum_{e=(s,t)} f_e = f.$$
$$\forall i \sum_{e=(j,i)} f_e - \sum_{e=(i,j)} f_e = 0.$$

Then, we get an equation for the flow variable, f_e's, we get the following equations

$$\forall e = (i, j), d_j - d_i + d_e \geq 0.$$

Except for arcs into the sink, with which we get

$$\forall e = (i, j), d_e - d_j \geq 0.$$

That is, we get that the d_j's are an upper bound on the distance to the sink.

Finally, we wish to minimize $\sum_e c_e d_e$, i.e., the coefficients here are the right hand sides of the primal.

For the variable f corresponding to flow out of the source, we get the equation

$$d_s \geq 1.$$

One solution to this dual problem is assigned $d_e = 1$ for all edges in the minimum cut. The value is equal to the cost of the minimum cut. Recall, that this is a lower bound on the value of the maximum flow.

Question: Is the value of the dual a lower bound on the value of the primal, in general?

Let's think of a feasible solution x. And consider $Ax - b$, which is positive. Let's multiply this vector by a feasible y of value yb.

We get $yAx - yb$ which must now be a positive value. But $yA \leq c$, and $yAx \leq cx$. That is, $cx - yb$ must be positive. That is the value of any feasible x, cx is at least as large as the value of any feasible y, yb.

This is called weak duality. (Working things out, this is a similar argument to showing that the minimum cut is a lower bound on the maximum flow.)

Exercise 2: Describe the primal and dual linear programs for the minimum cost flow problem.