Theory of Computing

Lecture 12

MAS 714

Hartmut Klauck

Linear Programming

- Linear Programming is the most powerful optimization problem that can be solved in polynomial time
  - Some generalizations [semidefinite programming] can be much more appropriate to use
- Fast LP solvers exist
  - Write an LP, solve it
  - Fast in theory/fast in practice...
- Important theoretical tool
- Duality

Linear Programming

- **Definition**
  - A Linear Program (LP) is an optimization problem over real variables $x_1,\ldots,x_n$
  - Maximizes/minimizes a linear function of x:
    max/min $C(x)=\sum_i c_i x_i$ (c_i are real coefficients)
  - Constraints: feasible solutions are those that satisfy a system of linear inqualities:
    - A is a real $m\times n$ matrix, b a vector
    - All x with $Ax \leq b$ are feasible
  - We are looking for a feasible x with maximum C(x)

Example: Shortest Path

- Variables: d(v) for all vertices v
- Objective function:

  Maximize -d(t)
- Constraints:

  $d(v) \leq d(u)+W(u,v)$ for all edges (u,v)

  $d(s)=0$

Example: Max Flow

- Graph G with capacities C(u,v)
- G has m edges (u,v), use m variables f(u,v)
- Inequalities:
  - $f(u,v) \leq C(u,v)$ for all edges (u,v)
  - $\sum_v f(u,v) = \sum_v f(v,u)$ for all u except s,t
  - $f(u,v) \geq 0$ for all edges
- Maximize $\sum_v f(s,v)$
- The program has m variables and m+n-2 inequalities/equations
  - Not counting nonnegativity constraint for f(u,v)
- By definition the maximum is a maximum flow

Standard form

- Constraints using $\geq$, $\leq$ and = are possible
- Easy to reduce to the *standard form*:
  - max c^Tx
  - Constraints:
  - $Ax \leq b$
  - $x \geq 0$
- c,b (column) vectors of lengths n,m
- x vector of n variables
- A is m by n matrix of coefficients

Duality

- Given an LP (A,b,c) in standard form
- We call this the *primal* LP
- The dual LP is defined as follow:
  - There are m variables y_i
  - Minimize $\sum_i b_i y_i$
  - Constraints:
    - $\sum_{i=1}^{m} A[i,j]\, y_i \geq c_j$
    - $y \geq 0$

Weak Duality

- **Claim:**
  - Let x be a feasible solution for the primal LP and y a feasible solution for the dual LP, then

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} b_i y_i$$

Proof

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} A[i,j] y_i \right) x_j$$

$$= \sum_{i=1}^{m} \left(\sum_{j=1}^{n} A[i,j] x_j \right) y_i$$

$$\leq \sum_{i=1}^{m} b_i y_i$$

Strong Duality

- Strong duality means that feasible solutions with the *same* value exist for the primal and the dual
- Compare Max Flow/Min Cut theorem

Nonstandard form

- The dual of a standard primal has a variable for each constraint of the primal and a constraint for each variable
  - And all variables are nonnegative
- If the primal has an equation $\sum a_i x_i = b$ then the dual has a variable which is not constrained to be nonnegative

Example

- Dual LP for Shortest Path
  - m nonnegative variables x(u,v) for edges (u,v)
  - n-1 constraints (for all vertices except s)
- Objective: Min $\sum_{(u,v)} x(u,v)W(u,v)$
- Constraints:
  - v not s,t: $\sum_{(u,v)} x(u,v) - \sum_{(v,u)} x(v,u) \geq 0$
  - $\sum_{(u,t)} x(u,t) - \sum_{(t,u)} x(t,u) \geq -1$
  - $x(u,v) \geq 0$

Interpretation

- Move one unit of flow (at least) from s to t
- Flow out of s unconstrained
- Each v other than s,t:
Outflow <= Inflow
- t:
Outflow<= Inflow – 1
- I.e., one unit of flow ``vanishes'' at t

Interpretation

- Cheapest way to achieve this:
  - Find a shortest path and flow 1 along it

Integrality

- Assume that all the edge weights are integers
- We know that the shortest path has integer cost
- And that there is a solution to the dual that simply puts $x(u,v)=1$ on the shortest path
- But will we find such a solution?
  - Or will the solution be something with noninteger $x(u,v)$?

Integrality

- Definition: a matrix M is totally unimodular, if the determinant of every square submatrix is either 0,1, or -1
- Theorem:
If the coefficient matrix of an LP is totally unimodular, then the Simplex algorithm will find an integer solution (b must be integer)
- For shortest path the matrix is totally unimodular

Example

- Dual LP for Max Flow
  - This will have m nonnegative variables [from m capacity constraints] and n-2 other variables [from n-2 flow conservation constraint]
  - Dual variables: x(u,v), y(v)
- Constraints:
  - There are m constraints
- Objective function:
  - min $\sum_{uv} x(u,v)\ C(u,v)$

Constraints

- One constraint for each edge (u,v)
  - f(u,v) appears in 3 constraints in the primal
- Constraints in the dual
    - y(s)=1 Note: y(s) and y(t) extra `variables’
    - y(t)=0
    - For each edge (u,v): $x(u,v) - y(u) + y(v) \geq 0$
    - For each edge (u,v): $x(u,v) \geq 0$
- Interpretation:
  - For all s-t- paths p: sum of weights x(u,v) on p is at least 1

Equivalence

- An s,t-cut (L,R) of capacity c yields a solution of cost c
  - Each edge from L to R receives weight 1
  - Set all y(v)=0
  - Cost is now equal to cut capacity
  - Constraints are true for every s,t path
- At least we know that the program is a relaxation of s,t-Min Cut
- How can we extract a cut from a solution of the dual LP?

Finding the cut

- First find shortest paths according to $x(u,v)$ using Dijkstra, call the distances $d(v)$
- $d(t) \geq 1$
- Let $L=\{v: d(v) \leq \Theta\}$ for some $\Theta \in [0,1)$
  - $s \in L$, $t \in R = V - L$
  - s,t, cut L,R
- Expected capacity for random Θ:
  - $\sum_{(u,v)} C(u,v)\ \mathrm{Prob}_{\Theta}\,[u \in L \text{ and } v \in R]$

Finding the cut

- $\sum_{(u,v)} C(u,v)\ \mathrm{Prob}_\Theta [u\in L \text{ and } v\in R]$
- But $\mathrm{Prob}_\Theta [u\in L \text{ and } v\in R] \leq d(v)-d(u) \leq x(u,v)$
  - The probability is the length of the interval [d(u),d(v)]
  - d(v)<=d(u)+x(u,v)
- Hence the expectation is at most $\sum_{(u,v)} C(u,v)\ x(u,v)$
- Which is the value of the program
- I.e., given a solution of the LP of cost at most c there is a cut L,R of cost at most c
- I.e., the optimum of the LP has the same cost as the Min st cut
- Finding the cut: exercise.

Linear Programming

- **Some facts:**
  - In practice LP‘s are usually solved with the Simplex Algorithm
  - Many math programs/libraries have LP solvers
  - Simplex is exponential in the worst case
  - There is a theoretical explanation why Simplex is fast in practice [smoothed analysis]
  - Worst case polynomial time algorithms:
    - Ellipsoid
    - Interior Point Methods