COL758: Advanced Algorithms **Spring 2019**

Lecture 6: January 17

Lecturer: Naveen Garg *Scribe: Paras Gupta*

Note: *LaTeX template courtesy of UC Berkeley EECS dept.*
Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

6.1 Multiplicative Update Method

The Multiplicative Update Method is motivated by the Expert Advice Method as seen in previous lectures and In this lecture we are going to solve the linear programming using the Multiplicative Update Method.

We saw the linear programming problem:

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

where $c, x \in \mathbb{R}^n,\ b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$.

Now What we will do, we will solve the problem in which that all the elements in the matrix A, b, c are positive.

$$\begin{aligned} a_{ij} &\geq 0 \\ b_i &\geq 0 \\ c_j &\geq 0 \end{aligned}$$

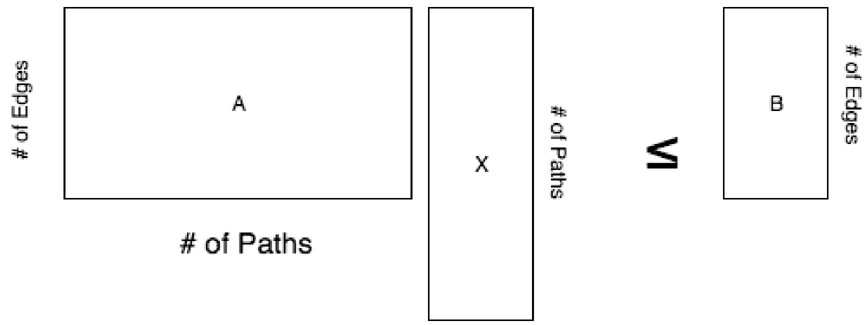
It is known as the **Packing Linear Problem**

6.2 Packing Linear Problem as the Max Flow Problem

Given a graph $G = (V, E)$ with edge capacities $c_e : E \rightarrow R^+$ and a source s and a sink t, the max flow problem is given by:

$$\begin{aligned} \max \quad & \sum_{i: P_i \in \P} f_i \\ \text{s.t.} \quad \forall e \in E \quad & \sum_{i: e \in P_i} f_i \leq c_e \\ \forall i : P_i \in P \quad & f_i \geq 0 \end{aligned}$$

The problem can be modelled as:



where $a_{ij} \in \{0, 1\}$ All paths j containing the edge i have $a_{ij} = 1$ as otherwise it is $a_{ij} = 0$.
X is the matrix for flow through the path and B is the capacity matrix.

But in the packing problem, there can be arbitrary value of a_{ij}.

6.3 Formulation of the problem

Now suppose, we have the following systme:

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & A_x \le b \\ & x \ge 0 \end{aligned}$$

But we have reduced our constraint to just single constraint. So, our problem will become:

$$\begin{aligned} \max \quad & \sum_j c_j x_j \\ \text{s.t.} \quad & \sum_j a_{1j} x_j \le b_1 \\ & \forall x_j \ge 0 \end{aligned}$$

The solution for the above problem will be given by the: Pick the j which maximizes $\frac{c_j}{a_{1j}}$

The above solution is intuitive and can be think of the solution of the knapsack problem which is equivalent to the above set of constraints.

Now, suppose we add one more constraint and have the equivalent version of the 2D Knapsack problem such that:

$$\begin{aligned} \max \quad & \sum_j c_j x_j \\ \text{s.t.} \quad & \sum_j a_{1j} x_j \le b_1 \\ & \sum_j a_{2j} x_j \le b_2 \\ & \forall x_j \ge 0 \end{aligned}$$

Now, as in the previous problem, we can say that its solution will be a linear combination and we have to pick the j which maximizes $\frac{c_j}{a_{1j}y_1+a_{2j}y_2}$
where $y_1 + y_2 = 1$, $0 \le y_1 \le 1 \quad 0 \le y_2 \le 1$

This is similar to picking shortest path. We have l_e with each path, we took linear combination and after that selected shortest among them. If we take the linear combination of rows and multiplying them by length and then column 1 will be the length of path 1, column 2 will be the length of path 2 and so on. So each column in that will give us the length of the respective path.

Suppose, If we have different profits on different paths and we have different length of paths. As in the previous algorithm, we changed the length of the edges to handle the congestion. # of paths using the edge reached near to the capacity thus giving us the optimum.

Thus the final model that we will be working with will be:

$$
\begin{aligned}
\max \quad & \sum_j c_j x_j \\
\text{s.t.} \quad & (\sum_j a_{1j} x_j \le b_1) \quad l_1 \\
& (\sum_j a_{2j} x_j \le b_2) \quad l_2 \\
& (\sum_j a_{3j} x_j \le b_3) \quad l_3 \\
& \quad . \\
& \quad . \\
& \quad . \\
& (\sum_j a_{m-1j} x_j \le b_{m-1}) \quad l_{m-1} \\
& (\sum_j a_{mj} x_j \le b_m) \quad l_m \\
& \forall x_j \ge 0
\end{aligned}
$$

6.4 Algorithm

1. Assign a length l_i with every column. Initialize l_i as : $l_i^0 = \frac{\delta}{b_i}$ where δ is an input parameter. Also set $x_j^*= 0$.
2. Repeat :
   (a) At each step we pick column j^* which maximizes $\frac{c_i}{\sum\limits_{i=1}^{m} a_{ij} l_i}$
   (b) Define $\lambda = \min\limits_i \frac{b_i}{a_{ij^*}}$
   (c) Updation step:-
   $\forall \quad 1 \le j \le n \quad x_{j^*} = x_{j^*} + \lambda$
   $\forall \quad 1 \le i \le m \quad l_i^t = l_i^{t-1}.(1 + \epsilon \frac{\lambda a_{ij^*}}{b_i})$

   until $D^t = 1$ where $D^t = \sum\limits_i l_i^t b_i$

Analysis

For every iteration $t \ge 1$, the value of objective of dual which is D^t is:

$$D^t = \sum_i l_i^t b_i$$

Substituting l_i^t from the updation step we get,

$$\begin{aligned} D^t &= D^{t-1} + \sum_i l_i^{t-1} (\epsilon \frac{\lambda^t a_{ij^*}}{b_i}) b_i \\ &= D^{t-1} + \epsilon \lambda^t \sum_i l_i^{t-1} a_{ij^*} \qquad (1) \\ &= D^{t-1} + \epsilon \lambda^t D^{t-1} \frac{c_{j^*}^t}{\beta} \qquad (2) \end{aligned}$$

Equation (2) is achieved by the use of dual problem:

Dual Problem is given by:

$$\begin{aligned} \min \quad & \sum_m b_i l_i \\ \text{s.t.} \quad & \sum_m a_{ij} l_i \ge c_j \\ & \forall \quad l_i \ge 0 \end{aligned}$$

If l_i is not feasible, we can get a feasible solution by scaling.

So, we will find the j^* which maximizes $\frac{c_j}{\sum_m a_{ij} l_i}$ that is which constraint is violated the most.

This will give us the feasible dual solution and since β is the minimum we get:

$$\sum_m b_i l_i \frac{c_{j^*}}{\sum_m a_{ij^*} l_i} \ge \beta \qquad (3)$$

Therefore,

$$D^t = D^{t-1}(1 + \epsilon\lambda^t \frac{c^t_{j^*}}{\beta})$$
$$\leq D^{t-1} e^{\epsilon \frac{\lambda^{t-1} c^t_{j^*}}{\beta}} \qquad \text{(Since } 1 + x \leq e^x\text{)}$$

Now let us say that after T iterations $D^T = 1$, then after T iterations,

$$D^T \leq D^0 e^{\frac{\epsilon}{\beta} \sum_{i=1}^{T} \lambda^i c^i_{j^*}} \qquad \text{(recursively)}$$

Observe that $\sum_{i=1}^{T} \lambda^i c_{j^*}$ is the primal objective value that is equal to $\sum_{j=1}^{n} x_j c_j$ in T iterations because λ^i is the value that we added in x^*_j in i^{th} iteration. Let this be denoted by P. Therefore by replacing the sum by P we get,

$$D^T \leq D^0 e^{\frac{\epsilon P}{\beta}}$$

Further by shifting terms and taking log both sides we get,

$$\frac{D^T}{D^0} \leq e^{\frac{\epsilon P}{\beta}}$$
$$\implies \frac{\epsilon P}{\beta} \geq \ln \frac{D^T}{D^0}$$

Note that $D^0 = m\delta$ where m is the number of constraints and $D^T = 1$ because of our stopping criterion. Therefore,

$$\frac{P}{\beta} \geq \frac{1}{\epsilon} \ln \frac{1}{m\delta} \qquad (3)$$

Claim: There is a feasible primal solution of value $\frac{P}{\frac{1}{\epsilon} \ln \frac{1}{\delta}}$

Proof: Consider a constraint i. For every b_i units increase in the LHS of the i^{th} constraint the corresponding dual variable l_i is increased by a factor of at least $1 + \epsilon$. Let us assume that $f_i = \sum_{j=1}^{n} a_{ij} x_j$ be the LHS of the i^{th} constraint. Therefore l_i increases by a factor of at least $(1 + \epsilon)^{\frac{f_i}{b_i}}$

Note that $l^0_i = \frac{\delta}{b_i}$ and $l^T_i \leq 1$ since $D^T = 1 = \sum_i l^T_i b_i$. Therefore,

$$(1 + \epsilon)^{\frac{f_i}{b_i}} \leq \frac{1}{\delta}$$

Further solving and taking log we get,

$$\frac{f_i}{b_i} \leq \frac{\ln \frac{1}{\delta}}{\ln (1 + \epsilon)}$$

From Taylor series expansion we can say that $\ln (1 + \epsilon) \approx \epsilon$ for small values of ϵ. Therefore we get,

$$\frac{f_i}{b_i} \leq \frac{1}{\epsilon} \ln \frac{1}{\delta}$$

Therefore every constraint is atmost violated by a factor $\frac{1}{\epsilon} \ln \frac{1}{\delta}$. Therefore scaling P by this quantity gives us the feasible primal solution of the packing LPP.

Value of primal therefore becomes $= \frac{P}{\frac{1}{\epsilon} \ln \frac{1}{\delta}}$

We want the primal solution to be close to the dual solution β. Therefore we want,

$$
\begin{aligned}
\frac{\beta}{P}\frac{1}{\epsilon}\ln\frac{1}{\delta} &\leq \frac{\frac{1}{\epsilon}\ln\frac{1}{\delta}}{\frac{1}{\epsilon}\ln\frac{1}{m\delta}} \leq 1+\gamma \\
\implies \frac{1}{\delta} &\leq (\frac{1}{m\delta})^{1+\gamma} = \frac{1}{m\delta}(\frac{1}{m\delta})^{\gamma} \\
\implies (\delta)^{\gamma} &\leq \frac{1}{m^{1+\gamma}} \\
\implies \delta &\leq \frac{1}{m^{1+\frac{1}{\gamma}}}
\end{aligned}
$$

We can thus choose δ accordingly. Observe the trade-off. We cannot choose small δ as then it will take more iterations to converge to the solution and we can also not choose δ large as we also got an upper bound on δ.