CS 598CSC: Approximation Algorithms	**Lecture date: February 18, 2011**
Instructor: Chandra Chekuri	**Scribe: CC**

In the previous lecture we discussed *packing* problems of the form $\max\{wx \mid Ax \leq 1, x \in \{0,1\}^n\}$ where A is a non-negative matrix. In this lecture we consider "congestion minimization" in the presense of packing constraints. We address a routing problem that motivates these kinds of problems.

1 Chernoff-Hoeffding Bounds

For the analysis in the next section we need a theorem that gives quantitative estimates on the probability of deviating from the expectation for a random variable that is a sum of binary random variables.

Theorem 1 (Chernoff-Hoeffding) *Let $X_1, X_2, \ldots, X_n$ be independent binary random variables and let $a_1, a_2, \ldots, a_n$ be coefficients in $[0,1]$. Let $X = \sum_i a_i X_i$. Then*

- *For any $\mu \geq \mathbb{E}[X]$ and any $\delta > 0$, $\Pr[X > (1+\delta)\mu] \leq \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\mu$.*
- *For any $\mu \leq \mathbb{E}[X]$ and any $\delta > 0$, $\Pr[X < (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$.*

The bounds in the above theorem are what are called *dimension-free* in that the dependence is only on $\mathbb{E}[X]$ and not on n the number of variables.

The following corollary will be useful to us. In the statement below we note that m is not related to the number of variables n.

Corollary 2 *Under the conditions of the above theorem, there is a universal constant α such that for any $\mu \geq \max\{1, \mathbb{E}[X]\}$, and sufficiently large m and for $c \geq 1$, $\Pr[X > \frac{\alpha c \ln m}{\ln \ln m} \cdot \mu] \leq 1/m^c$.*

Proof: Choose δ such that $(1+\delta) = \frac{\alpha c \ln m}{\ln \ln m}$ for some sufficiently large constant α that we will specify later. Let m be sufficiently large such that $\ln \ln m - \ln \ln \ln m > (\ln \ln m)/2$. Now applying the upper tail bound in the first part of the above theorem for μ and δ, we have that

$$\begin{aligned}
\Pr[X > \frac{\alpha c \ln m}{\ln \ln m} \cdot \mu] &= \Pr[X > (1+\delta)\mu] \\
&\leq \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\mu \\
&\leq \frac{e^\delta}{(1+\delta)^{(1+\delta)}} \quad \text{(since } \mu \geq 1 \text{ and the term inside is less than 1 for large } \alpha \text{ and } m) \\
&\leq \frac{e^{(1+\delta)}}{(1+\delta)^{(1+\delta)}} \\
&= (\frac{\alpha c \ln m}{e \ln \ln m})^{-\alpha c \ln m / \ln \ln m} \\
&= \exp((\ln \alpha c/e + \ln \ln m - \ln \ln \ln m)(-\alpha c \ln m / \ln \ln m)) \\
&\leq \exp(0.5 \ln \ln m(-\alpha c \ln m / \ln \ln m)) \quad (m \text{ and } \alpha \text{ are sufficiently large to ensure this}) \\
&\leq 1/m^{c\alpha/2} \leq 1/m^c \quad \text{(assuming } \alpha \text{ is larger than 2)}
\end{aligned}$$

□

2 Congestion Minimization for Routing

Let $G = (V, E)$ be a directed graph that represents a network on which traffic can be routed. Each edge $e \in E$ has a non-negative capacity $c(e)$. There are k pairs of nodes $(s_1, t_1), \ldots, (s_k, t_k)$ and each pair i is associated with a non-negative demand d_i that needs to be routed along a *single* path between s_i and t_i. In a first version we will assume that we are explicitly given for each pair i a set of paths $\mathcal{P}_i$ and the demand for i has to be routed along one of the paths in $\mathcal{P}_i$. Given a choice of the paths, say, $p_1, p_2, \ldots, p_k$ where $p_i \in \mathcal{P}_i$ we have an induced flow on each edge e. The flow on e is the total demand of all pairs whose paths contain e; that is $x(e) = \sum_{i:e \in p_i} d_i$. We define the congestion on e as $\max\{1, x(e)/c(e)\}$. In the congestion minimization problem the goal is to choose paths for the pairs to minimize the *maximum* congestion over all edges. We will make the following natural assumption. For any path $p \in \mathcal{P}_i$ and an edge $e \in p$, $c(e) \geq d_i$. One can write a linear programming relaxation for this problem as follows. We have variables $x_{i,p}$ for $1 \leq i \leq k$ and $p \in \mathcal{P}_i$ which indicate whether the path p is the one chosen for i.

$$
\begin{aligned}
& \min \lambda & \\
\text{subject to } & \sum_{p \in \mathcal{P}_i} x_{i,p} = 1 & 1 \leq i \leq k \\
\sum_{i=1}^{k} d_i & \sum_{p \in \mathcal{P}_i, e \in p} x_{i,p} \leq \lambda c(e) & e \in E \\
& x_{i,p} \geq 0 & 1 \leq i \leq k, p \in \mathcal{P}_i
\end{aligned}
$$

Technically the objective function should be $\max\{1, \lambda\}$ which we can enforce by adding a constraint $\lambda \geq 1$.

Let λ^* be an optimum solution to the above linear program. It gives a lower bound on the optimum congestion. How do we convert a fractional solution to an integer solution? A simple randomized rounding algorithm was suggested by Raghavan and Thompson in their influential work [1].

RandomizedRounding:
Let x be an optimum fractional solution
For $i = 1$ to k do
  Independent of other pairs, pick a single path $p \in \mathcal{P}_i$ randomly such that $\Pr[p \text{ is chosen}] = x_{i,p}$

Note that for a given pair i we pick exactly one path. One can implement this step as follows. Since $\sum_{p \in \mathcal{P}_i} x_{i,p} = 1$ we can order the paths in $\mathcal{P}_i$ in some arbitrary fashion and partition the interval $[0, 1]$ by intervals of length $x_{i,p}$, $p \in \mathcal{P}_i$. We pick a number θ uniformly at random in $[0, 1]$ and the interval in which θ lies determines the path that is chosen.

Now we analyze the performance of the randomized algorithm. Let $X_{e,i}$ be a binary random variable that is 1 if the path chosen for i contains the edge e. Let $X_e = \sum_i d_i X_{e,i}$ be the total demand routed through e. We leave the proof of the following claim as an exercise to the reader.

Claim 3 $\mathbb{E}[X_{e,i}] = \Pr[X_{e,i} = 1] = \sum_{p \in \mathcal{P}_i, e \in p} x_{i,p}$.

The main lemma is the following.

Lemma 4 *There is a universal constant* β *such that* $\Pr[X_e > \frac{\beta \ln m}{\ln \ln m} \cdot c(e) \max\{1, \lambda^*\}] \leq 1/m^2$ *where* m *is the number of edges in the graph.*

Proof: Recall that $X_e = \sum_i d_i X_{e,i}$.

$$\mathbb{E}[X_e] = \sum_i d_i \mathbb{E}[X_{e,i}] = \sum_i d_i \sum_{p \in \mathcal{P}_i, e \in p} x_{i,p} \leq \lambda^* c(e).$$

The second equality follows from the claim above, and inequality follows from the constraint in the LP relaxation.

Let $Y_e = X_e / c(e) = \sum_i \frac{d_i}{c(e)} X_{e,i}$. From above $\mathbb{E}[Y_e] \leq \lambda^*$. The variables $X_{e,i}$ are independent since the paths for the different pairs are chosen independently. Y_e is a sum of independent binary random variables and each coefficient $d_i/c(e) \leq 1$ (recall the assumption). Therefore we can apply Chernoff-Hoeffding bounds and in particular Corollary 2 to Y_e with $c = 2$.

$$\Pr[Y_e \geq \frac{2\alpha \ln m}{\ln \ln m} \max\{1, \lambda^*\}] \leq 1/m^2.$$

The constant α above is the one guaranteed in Corollary 2. We can set $\beta = 2\alpha$. This proves the lemma by noting that $X_e = c(e) Y_e$. □

Theorem 5 *RandomizedRounding, with probability at least* $(1 - 1/m)$ *(here* m *is the number of edges) outputs a feasible integral solution with congestion upper bounded by* $O(\frac{\ln m}{\ln \ln m}) \max\{1, \lambda^*\}$*.*

Proof: From Lemma 4 for any fixed edge e, the probability of the congestion on e exceeding $\frac{\beta \ln m}{\ln \ln m} \max\{1, \lambda^*\}$ is at most $1/m^2$. Thus the probability that it exceeds this bound for *any* edge is at most $m \cdot 1/m^2 \leq 1/m$ by the union bounds over the m edges. Thus with probability at least $(1 - 1/m)$ the congestion on all edges is upper bounded by $\frac{\beta \ln m}{\ln \ln m} \max\{1, \lambda^*\}$. □

The above algorithm can be *derandomized* but it requires the technique of *pessimistic estimators* which was another innovation by Raghavan [2].

2.1 Unsplittable Flow Problem: when the paths are not given explicitly

We now consider the variant of the problem in which $\mathcal{P}_i$ is the set of *all* paths between s_i and t_i. The paths for each pair are not explicitly given to us as part of the input but only implicitly given. This problem is called the *unsplittable flow problem*. The main technical issue in extending the previous approach is that $\mathcal{P}_i$ can be exponential in n, the number of nodes. We cannot even write down the linear proogram we developed previously in polynomial time! However it turns out that one can in fact solve the linear program implicitly and find an optimal solution which has the added bonus of having polynomial-sized support; in other words the number of variables $x_{i,p}$ that are strictly positive will be polynomial. This should not come as a surprise since the linear program has only a polynomial number of non-trivial constraints and hence it has an optimum basic solution with small support. Once we have a solution with a polynomial-sized support the randomized rounding algorithm can be implemented in polynomial time by simply working with those paths that have non-zero flow on them. How do we solve the linear program? This requires using the Ellipsoid method on the dual and then solving the primal via complementary slackness. We will discuss this at a later point.

A different approach is to solve a flow-based linear program which has the same optimum value as the path-based one. However, in order to implement the randomized rounding, one then has to decompose the flow along paths which is fairly standard in network flows. We now describe the flow based relaxation. We have variables $f(e,i)$ for each edge e and pair (s_i,t_i) which is the total amount of flow for pair i along edge e. We will send a unit of flow from s_i to t_i which corresponds to finding a path to route. In calculating congestion we will again scale by the total demand.

$$
\begin{aligned}
& \min \lambda \\
\text{subject to } & \sum_{i=1}^{k} d_i f(e,i) \leq \lambda c(e) && e \in E \\
\sum_{e\in\delta^+(s_i)} f(e,i) - \sum_{e\in\delta^-(s_i)} & f(e,i) = 1 && 1 \leq i \leq k \\
\sum_{e\in\delta^+(v)} f(e,i) - \sum_{e\in\delta^-(v)} & f(e,i) = 0 && 1 \leq i \leq k, v \notin \{s_i,t_i\} \\
& f(e,i) \geq 0 && 1 \leq i \leq k, e \in E
\end{aligned}
$$

The above linear program can be solved in polynomial time since it has only mk variables and $O(m+kn)$ constraints where m is the number of edges in the graph and k is the number of pairs. Given a feasible solution f for the above linear program we can, for each i, decompose the flow vector $f(.,i)$ for the pair (s_i,t_i) into flow along at most m paths. We then use these paths in the randomized rounding. For that we need the following flow-decomposition theorem for s-t flows.

Lemma 6 *Given a directed graph $G=(V,E)$ and nodes $s,t \in V$ and an s-t flow $f: E \to R_+$ there is a decomposition of f along s-t paths and cycles in G. More formally let $\mathcal{P}_{st}$ be the set of all s-t paths and let $\mathcal{C}$ be the set of directed cycles in G. Then there is a function $g: \mathcal{P}_{st} \cup \mathcal{C} \to R_+$ such that:*

- *For each e, $\sum_{q\in\mathcal{P}_{st}\cup\mathcal{C}, e\in q} g(q) = f(e)$.*
- *$\sum_{p\in\mathcal{P}_{st}} g(p)$ is equal to the value of the flow f.*
- *The support of g is at most m where m is the number of edges in G, that is, $|\{q | g(q) > 0\}| \leq m$. In particular, if f is acyclic $g(q) = 0$ for all $q \in \mathcal{C}$.*

Moreover, given f, g satisfying the above properties can be computed in polynomial time where the output consists only of paths and cycles with non-zero g value.

By applying the above ingredients we obtain the following.

Theorem 7 *There is an $O(\log n/\log\log n)$ randomized approximation algorithm for congestion minimization in the unsplittable flow problem.*

References

[1] P. Raghavan and C. D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica* 7(4):365–374, 1987.

[2] P. Raghavan. Probabilistic Construction of Deterministic Algorithms: Approximating Packing Integer Programs. *JCSS*, 37(2):130–143, 1988.