Randomized LP Rounding

1 Multiway Path Congestion

This lecture, we'll practice with a problem which applies all the knowledge we've gained so far, from LP relaxation and rounding to randomized algorithms and Chernoff bounds.

MULTIWAY PATH CONGESTION: Given k pairs (s_i, t_i) in a graph $\mathcal{G}$, find an (s_i, t_i) path for each i minimizing the maximum edge congestion (defined as the $\max_e$ number of paths using e).

For directed graphs, even $k = 2$ is **NP**-hard by a reduction from edge-disjoint paths. For undirected graphs, any constant k is solvable in polytime (exponential or worse in k); it is **NP**-hard if k is part of the input.

Let L represent the maximum load/congestion across all paths, and define indicator variables:

$$X_{i,P} = \begin{cases} 1 & \text{if } i \text{ uses path } P \\ 0 & \text{o.w.} \end{cases}$$

Now, we can represent the problem as an ILP:

Minimize

$$L$$

Subject to

$$\begin{cases} \sum_P X_{i,P} = 1 & \forall i \\ \sum_i \sum_{p \ni e} X_{i,P} \leq L & \forall e \\ X_{i,P} \geq 0 & \forall i, P \\ X_{i,P} \in \{0, 1\} & \forall i, P \end{cases}$$

To relax the ILP, we remove the integrality constraint. However, since there are exponentially many paths in $\mathcal{G}$, our ILP has exponentially many variables, which means we cannot solve this LP in polynomial time. Unless...

Proposition 1. *The LP has polynomially many nonzero variables.*

Proof. Notice that we can interpret $X_{i,P}$ as the amount of flow sent from s_i to t_i via P. Then, we have to send one unit of flow for each (s_i, t_i) pair. By binary search over L, we only need to check if L is feasible as a capacity for all edges, which is exactly MULTI-COMMODITY FLOW. By solving this problem, we obtain a valid (s_i, t_i) flow f_i for each i, and the smallest possible L for a fractional flow. Using a path decomposition of f_i, we get fractional variables $X_{i,P}$ solving the original LP, with at most m positive $X_{i,P}$. $\square$

Now that we have established a polytime fractional solution, we need a rounding algorithm. Notice that the $X_{i,P}$ for a fixed i sum to 1, so we interpret them as probabilities. Our rounding algorithm is to select the path P for the pair (s_i, t_i) with probability $X_{i,P}$ (we select exactly one P per i). Do this independently for each i. This clearly gives us a feasible solution, but we would like to analyze L.

Let L_e be the random variable representing the load on edge e (i.e., the number of paths using e). Then:

$$\mathbb{E}[L_e] = \mathbb{E}[\text{number of paths using } e]$$

$$= \sum_i \Pr[P_i \ni e]$$

$$= \sum_i \sum_{P \ni e} X_{i,P}$$

$$\leq L \text{ by the LP constraint}$$

We want a high-probability bound on $\max_e L_e$. So, we have to use tail bounds to show that:

$$\Pr[L_e > (1+\delta)L] \leq \frac{1}{m^2}$$

Then, we can take a union bound over m edges to show that we have a bounded $\max_e L_e$ with probability $\geq 1 - \frac{1}{m}$. To reverse-engineer the smallest possible δ, we can apply Chernoff bounds (which is legal because L_e is the sum of independent indicator variables). Because $\mathbb{E}[L_e] \leq L$:

$$\Pr[L_e \geq (1+\delta)L] < \frac{\exp(\delta)}{(1+\delta)^{(1+\delta)}}^L \leq \frac{1}{m^2}$$

Note that the LP has an integrality gap of 1 when it distributes everything fractionally. So, we strengthen the LP by adding the constraint of $L \geq 1$. That constraint simplifies our bound to:

$$\frac{\exp(\delta)}{(1+\delta)^{(1+\delta)}} \leq \frac{1}{m^2}$$

It is enough to set $\delta = \Theta(\frac{\log m}{\log \log m})$. Our output is a $\Theta(\frac{\log m}{\log \log m})$-approximation with probability $\geq 1 - \frac{1}{m}$, and if L is large, the bound gets even better.

2 Packet Routing

PACKET ROUTING: Given N pairs (s_i, t_i) with designated paths p_i. One packet each must be routed along p_i. Each edge can transmit at most one packet per timestep; the others wait in a buffer. We want to minimize the latest packet arrival.

Some apparent lower bounds are the maximum length of any path ($:= d$) and the maximum congestion of any edge ($:= c$). An upper bound on any reasonable algorithm is $\mathcal{O}(cd)$. There exists an algorithm which achieves $\mathcal{O}(c+d)$, but here we will prove an $\mathcal{O}(c + d\log(Nm))$-approximation with high probability.

To avoid frequent or repeated packet collisions, each packet selects independently and uniformly a start time $s_i \in \{1 \ldots r\}$ (we will figure out r soon). The packet waits for s_i steps, then tries to follow one edge in each step.

With randomly selected start times, there will still be some collisions, though much fewer. To handle them, we replace each step with a **meta-step** of length k real steps. In each meta-step, each packet traverses one edge; this works so long as $\leq k$ packets want to traverse any edge e in any one meta-step t. We represent this quantity with a random variable $X_{e,t}$ and define indicator variables:

$$X_{e,t_i} = \begin{cases} 1 & \text{if } i \text{ wants to use } e \text{ in meta-step } t \\ 0 & \text{o.w.} \end{cases}$$

Such that:

$$X_{e,t} = \sum_i X_{e,t_i}$$

We can now calculate the expectation:

$$\mathbb{E}[X_{e,t}] = \sum_i \Pr[i \text{ wants to use } e \text{ in meta-step } t]$$

$$\leq \sum_{i:e \in P_i} \frac{1}{r}$$

$$\leq \frac{c}{r}$$

Now, we want to find k such that $\Pr[X_{e,t} > k]$ is "small". Then, we will take a union bound over m edges and all $\leq d + r$ meta-steps, to obtain a failure rate of $\leq m^{-1}$. So, "small" in this case is $(m^2(d+r))^{-1}$. We have:

$$k = (1+\delta)\mathbb{E}[X_{e,t}]$$

With a fixed e, t, X_{e,t_i} are independent across i because the s_i are chosen independently. So, we can apply Chernoff bounds:

$$\Pr[X_{e,t} > k] < \frac{\exp(\delta)}{(1+\delta)^{(1+\delta)}}^{\frac{c}{r}}$$

This analysis will be continued in the next lecture.