TTIC 31150/CMSC 31150 Mathematical Toolkit (Fall 2024)

Avrim Blum

Lecture 13: Randomized Routing, Randomized Complexity Classes

Recap

- Basic tail inequalities: Markov's inequality and Chebyshev's inequality. Properties of variance: $Var(\sum_i X_i) = \sum_i Var(X_i)$ if pairwise independent. Threshold phenomena in random graphs.

- Chernoff-Hoeffding bounds: stronger bounds on large deviations using full mutual independence. For X a sum of independent Bernoulli R.V.s, we get:
  - $\mathbb{P}[X \geq (1+\delta)\mu] \leq \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}$
  - $\mathbb{P}[X \leq (1-\delta)\mu] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu}$

- For $\delta \in [0,1]$ get:
  - $\mathbb{P}[X \geq (1+\delta)\mu] \leq e^{-\delta^2\mu/3}$
  - $\mathbb{P}[X \leq (1-\delta)\mu] \leq e^{-\delta^2\mu/2}$

- Whp, poly(n) random vectors in $\{-1,1\}^n$ will all be nearly orthogonal. If toss n balls into n bins, whp no bin has $\gg \frac{\log n}{\log \log n}$ balls in it.

A small extension of Chernoff-Hoeffding bounds

- Suppose $X = X_1 + \cdots + X_n$ is a sum of independent Bernoulli(p_i) R.V.'s with $\mu = \mathbb{E}[X]$.
- Suppose we have an upper-bound B on μ (i.e., $\mu \leq B$).
- Then we can say: $\mathbb{P}[X \geq (1+\delta)B] \leq e^{-\delta^2 B/3}$. [I.e., we can use B in exponent]

Analysis:

We can do this so long as $B \leq n$. If $B > n$ then the bound holds trivially.

- Define $p_1', \ldots, p_n' \in [0,1]$ such that $p_i' \geq p_i$ and $\sum_i p_i' = B$.
- Define R.V. X_i': if $X_i = 1$ then $X_i' = 1$; else if $X_i = 0$ then $X_i' = 1$ with prob $\frac{p_i' - p_i}{1 - p_i}$.
- The X_i' are independent Bernoulli(p_i') R.V.s, so $\mathbb{P}[\sum_i X_i' \geq (1+\delta)B] \leq e^{-\delta^2 B/3}$.
- But notice that $\sum_i X_i' \geq \sum_i X_i$ always. So, our desired inequality holds too.

Low-congestion routing

Given a directed graph G and a collection of pairs of vertices $\{(s_i, t_i)\}$, we would like to route paths from s_i to t_i to minimize the maximum congestion (the number of paths using any given edge).

This problem is NP-hard. Can we get a good approximation?

Raghavan & Thompson idea

- First solve the problem fractionally (also called "multi-commodity flow"):
  - For each (directed) edge (u, v) and each commodity i, have variable $x_{i,(u,v)}$.
  - For each i have constraints: $\sum_v x_{i,(s_i,v)} = 1, \sum_u x_{i,(u,t_i)} = 1$, and flow-in = flow-out for all $v \notin \{s_i, t_i\}$: $\sum_u x_{i,(u,v)} = \sum_{u'} x_{i,(v,u')}$. Also, non-negativity.
  - Then for each edge (u, v) have constraint $\sum_i x_{i,(u,v)} \leq C$ and minimize C.
- Note that if opt is the value of the optimal solution to the original problem, then $C \leq opt$, because this is a relaxation. But now we have to convert our flow into a collection of s_i-t_i paths.

Raghavan & Thompson idea

- First solve the problem fractionally (also called "multi-commodity flow"):
  - For each (directed) edge (u, v) and each commodity i, have variable $x_{i,(u,v)}$.
  - For each i have constraints: $\sum_v x_{i,(s_i,v)} = 1, \sum_u x_{i,(u,t_i)} = 1$, and flow-in = flow-out for all $v \notin \{s_i, t_i\}$: $\sum_u x_{i,(u,v)} = \sum_{u'} x_{i,(v,u')}$. Also, non-negativity.
  - Then for each edge (u, v) have constraint $\sum_i x_{i,(u,v)} \leq C$ and minimize C.
- Next, for each i, we view the values $x_{i,(u,v)}$ as probabilities and select a path from s_i to t_i such that for each (u, v), $\mathbb{P}[(u, v) \text{ is selected}] = x_{i,(u,v)}$.
  - Claim: we can do this by starting from s_i and choosing an outgoing edge with probability proportional to the flow of commodity i on that edge, continuing until t_i is reached.

Raghavan & Thompson idea

- First solve the problem fractionally (also called "multi-commodity flow"):
  - For each (directed) edge (u, v) and each commodity i, have variable $x_{i,(u,v)}$.
  - For each i have constraints: $\sum_v x_{i,(s_i,v)} = 1, \sum_u x_{i,(u,t_i)} = 1$, and flow-in = flow-out for all $v \notin \{s_i, t_i\}$: $\sum_u x_{i,(u,v)} = \sum_{u'} x_{i,(v,u')}$. Also, non-negativity.
  - Then for each edge (u, v) have constraint $\sum_i x_{i,(u,v)} \leq C$ and minimize C.
- Next, for each i, we view the values $x_{i,(u,v)}$ as probabilities and select a path from s_i to t_i such that for each (u, v), $\mathbb{P}[(u, v)$ is selected$] = x_{i,(u,v)}$.
  - Proof: Consider the DAG of flows of commodity i. Argue by induction on this DAG, using the flow-in = flow out constraint.

Raghavan & Thompson idea

- First solve the problem fractionally (also called "multi-commodity flow"):
- Next, for each i, we view the values $x_{i,(u,v)}$ as probabilities and select a path from s_i to t_i such that for each (u,v), $\mathbb{P}[(u,v) \text{ is selected}] = x_{i,(u,v)}$.

Claim: If $opt \gg \log n$ then whp this will find a solution of max congestion $\leq \left(1 + o(1)\right) \cdot opt$.
For any value of opt, whp this will find a solution of congestion $O\left(\frac{\log n}{\log \log n} \cdot opt\right)$.

Proof:

- Let $X_{i,(u,v)}$ be an indicator R.V. for the event that we use edge (u,v) in the s_i-t_i path.
- $\mathbb{E}[X_{i,(u,v)}] = x_{i,(u,v)}$, and $X_{1,(u,v)}, X_{2,(u,v)}, \ldots$ are independent for any given (u,v).
- So, we can apply Chernoff-Hoeffding to $X_{(u,v)} = \sum_i X_{i,(u,v)}$, where $\mathbb{E}\left[X_{(u,v)}\right] \leq opt$.

Raghavan & Thompson idea

- $\mathbb{P}\left[X_{(u,v)} \geq (1+\delta)opt\right] \leq e^{-\delta^2\, opt/3}$. If $opt \gg \log n$, the RHS is $o(1/n^2)$ for any constant $\delta > 0$, so the chance there exists an edge with greater congestion is $o(1)$.

Claim: If $opt \gg \log n$ then whp this will find a solution of max congestion $\leq \big(1+o(1)\big) \cdot opt$.
For any value of opt, whp this will find a solution of congestion $O\left(\frac{\log n}{\log \log n} \cdot opt\right)$.

Proof:

- Let $X_{i,(u,v)}$ be an indicator R.V. for the event that we use edge (u, v) in the s_i-t_i path.
- $\mathbb{E}[X_{i,(u,v)}] = x_{i,(u,v)}$, and $X_{1,(u,v)}, X_{2,(u,v)}$, ... are independent for any given (u, v).
- So, we can apply Chernoff-Hoeffding to $X_{(u,v)} = \sum_i X_{i,(u,v)}$, where $\mathbb{E}\left[X_{(u,v)}\right] \leq opt$.

Raghavan & Thompson idea

- $\mathbb{P}\left[X_{(u,v)} \geq (1+\delta)opt\right] \leq e^{-\delta^2\, opt/3}$. If $opt \gg \log n$, the RHS is $o(1/n^2)$ for any constant $\delta > 0$, so the chance there exists an edge with greater congestion is $o(1)$.

Claim: If $opt \gg \log n$ then whp this will find a solution of max congestion $\leq \left(1+o(1)\right) \cdot opt$.
For any value of opt, whp this will find a solution of congestion $O\left(\frac{\log n}{\log\log n} \cdot opt\right)$.

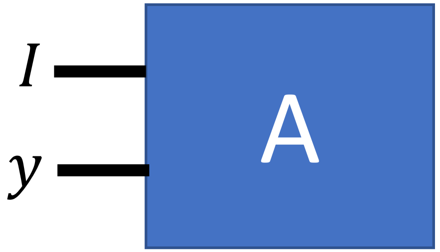
Proof:

- For any value of opt, can use $\mathbb{P}\left[X_{(u,v)} \geq k\ opt\right] < \left(\frac{e^{k-1}}{k^k}\right)^{opt} \leq \frac{e^{k-1}}{k^k}$. Set $k = \frac{3\ln n}{\ln\ln n}$ and get $o(1/n^2)$ as desired.

Randomized Complexity Classes

- Introduce **RP** and **BPP**, which are randomized versions of complexity class **P**.
- Formally, considering decision (YES/NO) problems. E.g., "does the given graph G have a perfect matching?"
- Definition: An algorithm runs in **polynomial time** if for some constant c, its running time on instances of size n is $O(n^c)$.
- Definition: **P** is the class of decision problems solvable by deterministic polynomial-time algorithms.

To define randomized complexity classes, will consider algorithms A that take in *two* inputs: an instance I and an auxiliary input y, which is a bit string of length polynomial in the size of I. Think of y as the random bits used by A.



Randomized Complexity Classes

- Definition: A problem Q is in **RP** if there exists a polynomial-time algorithm $A(I, y)$ and a polynomial r such that:
  - If I is a YES-instance then $\mathbb{P}_{y \in \{0,1\}^{r(|I|)}}[A(I, y) = YES] \geq \frac{1}{2}$.
  - If I is a NO-instance then $\mathbb{P}_{y \in \{0,1\}^{r(|I|)}}[A(I, y) = YES] = 0$.

RP corresponds to problems solvable by randomized algorithms with 1-sided error.

E.g., we showed Perfect Matching ∈ **RP** because we gave an algorithm such that if G has a perfect matching, then the algorithm says YES with probability ≥ ½ (because the Tutte polynomial is not identically 0), and if G does not have a perfect matching, then the algorithm is guaranteed to say NO.

Randomized Complexity Classes

- Definition: A problem Q is in **BPP** if there exists a polynomial-time algorithm $A(I, y)$ and a polynomial r such that:
  - If I is a YES-instance then $\mathbb{P}_{y \in \{0,1\}^{r(|I|)}}[A(I, y) = YES] \geq \frac{3}{4}$.
  - If I is a NO-instance then $\mathbb{P}_{y \in \{0,1\}^{r(|I|)}}[A(I, y) = YES] \leq \frac{1}{4}$.

BPP corresponds to randomized algorithms with 2-sided error.

It is believed that **P=RP=BPP**, but there is no deterministic polynomial-time algorithm known for the polynomial identity-testing problem.

One more interesting complexity class to mention, **P/poly**, which is the class of problems solvable in "non-uniform polynomial time".

Randomized Complexity Classes

- Definition: A problem Q is in **P/poly** if there exists a polynomial-time algorithm $A(I, y)$ and a polynomial r such that for every n there exists a string $y_n \in \{0,1\}^{r(n)}$ such that $A(I, y_{|I|})$ is always correct.

Think of y_n as an "advice" string for inputs of size n.

There could be 2^n inputs of size n, but y_n has size only $r(n)$, so it can't just encode all the answers.

Claim: **RP** $\subseteq$ **P/poly**. (You will show **BPP** $\subseteq$ **P/poly** on your homework).

Proof: Suppose $Q \in$ **RP**. So, there exists algo A and polynomial r satisfying **RP** definition.

- Define A' that on instance I of size n uses auxiliary input y_n of length $(n+1)r(n)$ to perform $n+1$ runs of A and output YES if any run gives YES, else NO.
- $\mathbb{P}_{y_n}[A'(I, y_n)$ fails$] \leq 1/2^{n+1}$.
- $\mathbb{P}_{y_n}[$exists I of size n s.t. $A'(I, y_n)$ fails$] \leq \frac{2^n}{2^{n+1}} = \frac{1}{2}$. So, a good y_n must exist.