Mathematics for Computer Science (30470023) 2020 Spring

Lecture 5 — March 16, 2020

Instructor: Andrew C. Yao *Scribes: Wu Mengdi, Hu Zihan*

1 Overview

In this lecture we will finish our introduction to the basic tools of probability and show how these can be applied to real examples. In the first part, we will introduce the notion of independent random variables, and then give a proof of Chernoff inequality. In the second part, we will discuss the Network Routing Problem.

2 Independent random variables

Definition 2.1 (Mutually independent). Random variables $X_1, X_2, \ldots, X_n$ are mutually independent (all independent) if

$$\Pr\{X_1 = a_1, X_2 = a_2, \ldots, X_n = a_n\} = \prod_{i=1}^{n} \Pr\{X_i = a_i\}$$

for all $a_1, a_2, \ldots, a_n$. ♢

Definition 2.2 (K-wise independent). Random variables $X_1, X_2, \ldots, X_n$ are k-wise independent if

$$\Pr\{X_{i_1} = a_{i_1}, X_{i_2} = a_{i_2}, \ldots, X_{i_k} = a_{i_k}\} = \prod_{j=1}^{k} \Pr\{X_{i_j} = a_{i_j}\}$$

for all $i_1, i_2, \ldots, i_k$ and $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$. ♢

Remark 2.3. If $k = 2$, we say they are pairwise independent.

Theorem 2.4. If $X_1, X_2, \ldots, X_n$ are mutually independent, then

$$\mathbb{E}(\prod_{i=1}^{n} X_i) = \prod_{i=1}^{n} \mathbb{E}(X_i).$$ ♢

Proof. Straightforward. □

Theorem 2.5. If $X_1, X_2, \ldots, X_n$ are pairwise independent, then

$$\mathrm{Var}(\sum_{i=1}^{n} X_i) = \sum_{i=1}^{n} \mathrm{Var}(X_i).$$

♢

Proof. Straightforward. □

Example 2.6. Throrem 2.5 gives us a second way to compute the variance of n coin tosses. Recall that in the 0-1 tosses example, the random variables X_i (the result of the i-th toss) are all independent. Therefore we have

$$\mathrm{Var}(X) = \sum_{i=1}^{n} \mathrm{Var}(X_i) = n\mathrm{Var}(X_1) = nb(1-b),$$

which agrees with our conclusion in the last lecture.

3 Chernoff inequality

Chernoff inequality gives bounds of tail probability.

Theorem 3.1 (Chernoff inequality). Suppose $X_1, X_2, \ldots, X_n$ are independent random variables such that $X_i \in \{0,1\}$ and $\Pr\{X_i = 1\} = p_i$ for $1 \leq i \leq n$, and $X = \sum_i X_i$. Let $\mu = \mathbb{E}(X) = \sum_i p_i$. Then

$$\Pr\{X > (1+\delta)\mu\} \leq \left(\frac{\mathrm{e}^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}, \delta > 0$$
$$\Pr\{X < (1-\delta)\mu\} \leq \left(\frac{\mathrm{e}^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu}, 0 < \delta < 1. \qquad \diamondsuit$$

Proof. By Markov's inequality, for $t > 0$,

$$\Pr\{X > (1+\delta)\mu\} \leq \Pr\{\mathrm{e}^{tX} > \mathrm{e}^{t(1+\delta\mu)}\} \leq \frac{\mathbb{E}(\mathrm{e}^{tX})}{\mathrm{e}^{t(1+\delta\mu)}}.$$

Based on this, we find the bound by two steps:

- Step 1: Obtain the right-hand side as a function of t;
- Step 2: Pick the best t to optimize the bound.

Let's do the first step.

$$\begin{aligned}
\mathbb{E}(\mathrm{e}^{tX}) &= \mathbb{E}(\mathrm{e}^{t\sum_i X_i}) \\
&= \prod_i \mathbb{E}(\mathrm{e}^{tX_i}) \\
&= \prod_i (p_i\mathrm{e}^t + (1-p_i)) \\
&= \prod_i (1 + p_i(\mathrm{e}^t - 1)) \\
&\leq \prod_i \mathrm{e}^{p_i(\mathrm{e}^t-1)} \\
&= \mathrm{e}^{(\mathrm{e}^t-1)\sum_i p_i} \\
&= \mathrm{e}^{\mu(\mathrm{e}^t-1)}
\end{aligned}$$

and then

$$\Pr\{X > (1+\delta)\mu\} \le \frac{\mathbb{E}(\mathrm{e}^{tX})}{\mathrm{e}^{t(1+\delta)\mu}} \le \mathrm{e}^{-t(1+\delta)\mu+\mu(\mathrm{e}^t-1)}. \tag{1}$$

Now we get the function $f(t) = \mathrm{e}^{-t(1+\delta)\mu+\mu(\mathrm{e}^t-1)}$ and the second step is to choose a value of t to minimize $f(t)$. This can be easily done by differentiation, and the best choice is

$$t = \ln(1+\delta). \tag{2}$$

Substituting (2) into (1), we finally get the bound

$$\Pr\{X > (1+\delta)\mu\} \le \left(\frac{\mathrm{e}^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}, \delta > 0.$$

Proof for the second inequality is similar, and it will be left as an exercise in the homework. □

4 Application: Network Routing Problem

Some users or computers are connected by a network. When they send messages (packets) to each other, there will be congestion. How to rout the messages properly to ease the congestion so as to reduce the delay?

4.1 The Model

We establish a model of hypercube network.

Suppose there are $N = 2^n$ nodes $0, 1, 2, \ldots, 2^n - 1$. Two nodes i and j is connected if and only if the Hamming distance of their binary representation is 1.

At the beginning, every node $i \in \{0,1\}^n = \{0, 1, 2, \ldots, N-1\}$ has a packet M_i which i wants to send to node $\sigma(i) \in \{0,1\}^n$. Suppose no one receives two packets, then we have that σ is a permutation.

At each step, all M_i can move along an edge or not move. M_i can move from node k to node j if and only if k, j is connected and there is no other message which moves from node k to node j at that step.

Our goal is to find a way to minimize the number of steps to take.

4.2 The First Try: Bit-Fixing Routing Algorithm

Denote the path that M_i takes as $Path_i$. Let's consider a simple way — just greedily decrease the Hamming distance. That is to say, we just compare i and $\sigma(i)$, and then change the first bit of i which is different from $\sigma(i)$, and then change the second bit of i which is different from $\sigma(i)$, etc. We record it as $Path_i$. For example, if $i = (0011101)_2$ and $\sigma(i) = (0110001)_2$, then the nodes along $Path_i$ will be $\{(0111101)_2, (0110101)_2, (0110001)_2\}$.

Clearly, $|Path_i| = d_H(i, \sigma(i))$ (the Hamming distance between i and $\sigma(i)$) Therefore, $|Path_i| \leq n$ and there exists a node i and a permutation σ such that $|Path_i| = n$, which indicates that there exists a permutation σ such that there is no way of sending all the messages in less than n steps.

However, does there exists a permutation σ such that some message will have a very long delay using Bit-Fixing Routing Algorithm?

The answer is yes. When n is even, say $n = 2k$. We can establish a permutation σ satisfying $\forall u \in \{0,1\}^{k-1}, \sigma((0\ u\ 0^k)_2) = (0^k\ 1\ u)_2$ Then for all node $i = (0\ u\ 0^k)_2, M_i$ will move from $(0^{2k})_2$ to $(0^k\ 1\ 0^{k-1})_2$ at some step in the procedure. However, at each step, only one message is allowed to use the edge. Therefore, there is at least $2^{k-1} = \Theta(\sqrt{N})$ steps.

In fact, there is a related theorem.

Definition 4.1. An oblivious routing algorithm is an algorithm such that all $Path_i$ only depends on i and $\sigma(i)$(not depend on $\sigma(j), j \neq i$). ♢

Theorem 4.2 (Borodin and Hopcroft). Any deterministic oblivious routing algorithm has $\sqrt{\frac{N}{d}}$ delay time. ♢

4.3 Another Try

Are there better ways? How to avoid that constraints?

Theorem 4.3 (Brenda and L. Valiant 1981). There is a randomized routing algorithm that delivers every M_i to its destination in $O(n)$ steps with probability $1 - 2^{-cn}$ for some constant $c > 0$. ♢

The algorithm performs in two phases:

Phase I: For each $i \in \{0, 1, \ldots, N-1\}$, choose a random node $\rho(i)$ from $\{0, 1, \ldots, N-1\}$ uniformly. Then we send all M_i to $\rho(i)$ by Bit-Fixing. Wait for $7n$ steps.

Phase II: Let all M_i go from $\rho(i)$ to $\sigma(i)$ by Bit-Fixing. Wait for another $7n$ time steps.

Definition 4.4. Delay of a message M_i is defined as (the number of steps used - $|Path_i|$). ♢

Lemma 4.5 (Main Lemma). For each $i \in \{0,1\}^n$,

$$\Pr\{\text{delay of } M_i > 6n\} \leq 2^{-4n}$$

In order to prove it, we need another lemma.

Lemma 4.6 (Main Lemma'). Fix $i \in \{0, 1, \ldots, N-1\}$ and $\rho(i) \in \{0, 1, \ldots, N-1\}$, then we randomly picks $\rho(j)$ for all $j \neq i$, then we have that

$$\Pr\{\text{delay of } M_i > 6n\} \leq 2^{-4n}$$

Proof. Let $S = \{j \neq i \mid Path_i \cap Path_j \neq \emptyset\}$. Note that the delay of $M_i \leq |S|$.

$$\begin{aligned}
E(\text{delay of } M_i) \leq E(|S|) &= \sum_{j \neq i} \Pr\{Path_i \cap Path_j \neq \emptyset\} \\
&\leq \sum_{j \neq i} \sum_{k=1}^{n} \Pr\{j \equiv i \mod 2^{n-k}, [\frac{\rho(j)}{2^{n-k+1}}] = [\frac{\rho(i)}{2^{n-k+1}}], [\frac{j}{2^{n-k}}] \not\equiv [\frac{\rho(j)}{2^{n-k}}] \mod 2\} \\
&\leq \sum_{k=1}^{n} 2^k \frac{1}{2^k} \\
&= n
\end{aligned}$$

Let X_j be the random variable such that $X_j = 1$ if and only if $Path_i \cap Path_j \neq \emptyset$.

By Chernoff Inequality,

$$\Pr\{\text{delay of } M_i > 6n\} \leq \Pr\{|S| > 6n\} = \Pr\{\sum_j X_j \geq 6n\} \leq 2^{-4n} \qquad \square$$

Now let's prove Main Lemma.

Proof. For each $i \in \{0,1\}^n$,

$$\Pr\{\text{delay of } M_i > 6n\} = \sum_{k=0}^{N-1} \Pr\{\rho(i) = k\} \Pr\{\text{delay of } M_i > 6n | \rho(i) = k\} \leq N \frac{1}{N} 2^{-4n} = 2^{-4n} \qquad \square$$

Now let's prove the algorithm delivers every M_i to its destination in $O(n)$ steps with probability $1 - 2^{-cn}$ for some constant $c > 0$

Proof. Using Main Lemma, we know that

$$\Pr\{\exists i, M_i \text{ fails to arrive } \rho(i) \text{ in the first Phase}\} \leq \sum_{i=0}^{N-1} \Pr\{\text{delay of } M_i > 6n\} \leq N \times 2^{-4n} = 2^{-3n}$$

The second Phase is just like the inverse of the first Phase.

Therefore, we have that

$$\Pr\{\exists i, M_i \text{ fails to arrive } \sigma(i) \text{ in the second Phase}\} \leq 2^{-3n}$$

As a result, this algorithm delivers every M_i to its destination in $14n = O(n)$ steps with probability $\geq (1 - 2^{-3n})^2 \geq 1 - 2^{-3n+1} \geq 1 - 2^{-2n}$. $\square$