Probability in Computing
Basic Tools & Algorithms

Contents

1 Introduction

An Example: Verifying polynomial multiplication

Is $(x^2 - x + 3)(x^3 + 4x^2 - x - 1) = x^5 + 3x^4 - x^3 + 12x^2 - 3x - 3$?

Given three polynomials $f(x), g(x), h(x) \in \mathbb{Z}[x]$, how can we efficiently verify whether $f(x)g(x) = h(x)$? The direct method is to just multiply $f(x)$ and$g(x)$ in time $O(n \log n)$ (using FFT) and compare it with $h(x)$; this can be done in time $O(n \log n)$, where $n = deg(h)$.

Using randomness, we can verify this in $O(n)$ time, provided we allow for a small probability that our algorithm will output an incorrect answer.

Here's an algorithm: pick uniformly at random, an element $a \in \{1, 2, \ldots, 100n\}$; evaluate $f(a), g(a), h(a)$ in $O(n)$ time, and compare the product of the first two values with the third. If they match, we output TRUE, otherwise we output FALSE.

Notice that if it were indeed true that $f(x)g(x) = h(x)$, our values would match, and we shall output the correct answer (TRUE). On the other hand, it can happen that $f(x)g(x) \neq h(x)$, but $f(a)g(a) = h(a)$ for the value a that we picked. In this case, the correct answer is FALSE, but our output is TRUE.

What is the probability that we give this incorrect answer? To answer this, suppose that $f(x)g(x) \neq h(x)$, and let $S = \{a | f(a)g(a) = h(a)\}$. We note that the non-zero polynomial $f(x)g(x) - h(x)$ can have at most n roots, and therefore $|S| \leq n$. Now, the probability of our giving the incorrect answer is exactly equal to $\frac{|S \cap \{1, 2, \ldots, 100n\}|}{100n}$, which is at most $\frac{1}{100}$.

Thus, we have a (simple) randomized algorithm that verifies the polynomial-multiplication identity in $O(n)$ time, and which returns the correct answer with probability at least 0.99.

Exercise 1 *Substitute random values for the polynomials at the beginning of this section and compare. Are they equal?*

1.1 Monte Carlo and Las Vegas algorithms

Randomized algorithms often offer an advantage in speed or simplicity (or both). The random choices that the algorithm makes can result in one or both of the following:

(A) the algorithm does not always return the correct value: instead, it returns the correct value with some probability;

(B) the running time is a random variable.

Algorithms with property (A) are called Monte Carlo algorithms, and algorithms with property (B) are called Las Vegas algorithms. The first algorithm we saw in Section 1 is an example of a Monte Carlo algorithm. The most well-known Las Vegas algorithm is quick-sort which always returns the sorted sequence correctly, but its running time depends on the choice of random pivots, and is therefore a random variable.

1.2 Search-vs-decision

A decision problem is one whose answer is True/False, such as verifying a polynomial identity, or whether a CNF formula has a satisfying assignment.

A search problem, on the other hand, is about finding an object with some property, and in optimization problems, the property is to minimize or maximize some objective function.

For a Monte-Carlo algorithm for these two kinds of problems, what is a good probablity of correctness/success respectively? As the exercises below show, the answer isn't the same for both!

Exercise 2 *For a Monte-Carlo algorithm that solves a decision problem, what is a good value for the probability of correctness?*

Exercise 3 *For a Monte-Carlo algorithm that solves a search problem, what is a good value for the probability of correctness?*

The two exercises leave open-ended the meaning of "good", but let's say that 0.99 is certainly a good value. Let's also say that an algorithm is efficient if its worse-case running time is polynomial in the size of the input, and in both exercises, the algorithms are efficient.

1.3 Basic probability review

Some exercises from class:

1. A number X is chosen uniformly at random from $\{1, 2, 3, \ldots, 100\}$. What is the probability that X is divisible by 2 or by 3?

   Solution: Let A be the event that X is divisible by 2, B the event that X is divisible by 3. We are asking for $Pr[A \cup B] = Pr[A] + Pr[B] - Pr[A \cap B] = \frac{50}{100} + \frac{33}{100} - \frac{16}{100} = \frac{67}{100}$.

2. A fair coin is tossed 100 times. What is the probability that the tosses result in at least 70 heads?

   Solution: The direct expression is $\frac{1}{2^{100}}\left(\sum_{i=70}^{100}\binom{100}{i}\right)$. Note, however that from this expression we are unable to gauge how small or large the actual value is. As some-one in class pointed out, an upper bound is $\frac{30\binom{100}{70}}{2^{100}}$.

3. A pair of 6-sided dice is thrown. Given that the sum of the observed values is at least 10, what is the probability that the first value is even?

   Solution: The reduced sample space is $\{(4,6),(6,4),(5,5),(5,6),(6,5),(6,6)\}$ and the desired probability is $\frac{4}{6}$.

Abstract definition of probability:

Given any set Ω, the universe (or referred to as the sample space) or universse, a probability distribution on Ω is a function P from subsets of Ω to $[0,1]$, that satisfies the following properties:

(a) $P(\Omega) = 1$; (b) If $E_1, E_2, \ldots$ are mutually disjoint, then $P(E_1 \cup E_2 \ldots) = P(E_1) + P(E_2) + \ldots$.

An *event* is any subset of Ω.

Three further useful properties are:

1. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.
2. For an event E, we have $P(\bar{E}) = P(\Omega \setminus E) = 1 - P(E)$.
3. If A, B are two independent events, then $P(A \cap B) = P(A)P(B)$.

Conditional probability: If $A, B \subseteq \Omega$ are two events, the conditional probability $P(A|B)$ is obtained by restricting the sample space to B. Thus, we have:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

which is the definition of $P(A|B)$.

1.4 Sampling with a coin

Suppose that we have a fair coin as a source of randomness. We can use it to sample from the uniform distribution on $\{1, 2, \ldots, 8\}$ by tossing it three times, and assigning the values 1 through 8 to the eight possible outcomes. More generally, we can also sample from a set such as $\{1, 2, \ldots, n\}$ by tossing it k times for k such that $2^k \geq n$, and assigning n distinct outcomes the values 1 through n. If the outcome is not one of these n, we repeat the experiment.

In particular, we can also make choices with probability p when p is a rational number. We will also think of this as having a biased coin with probabilities p and $1 - p$ for the two sides.

Exercise 4 *Simulate a coin of any real bias p using a fair coin.*

Exercise 5 *Simulate a fair coin using a biased coin of unknown bias.*

2 Minimum Cut

For an undirected graph $G = (V, E)$, a cut is a partition $(A, V \setminus A)$ of the vertices, and the set of edges $\{\{x, y\} : x \in A, y \in V \setminus A\}$ is called a cut-set, and sometimes the edge-set is also referred to as the cut, as we shall do. An example is shown in the slides.

A Greedy Algorithm that doesn't work:

We maintain sets of vertices, with each set initially consisting of singleton vertices, and repeatedly merge them, until we have exactly two sets remaining; we then output the set of edges across these two sets as the solution.

Which pair of sets should we merge at any time-step? When there are three sets remaining, we can see that we should merge the pair that has the maximum number of edges across it. For more sets, however, the greedy algorithm can give an incorrect answer.

2.1 Karger's Algorithm

The algorithm of David Karger is similar to the greedy algorithm, in the sense that we start with singleton sets, and then repeatedly merge them. However, instead of doing this greedily, Karger's algorithm merges each pair of sets with probability proportional to the number of (multi)-edges between them.

An example of the merging process is shown in the slides, where we think of each set as a single (block) vertex, and have multi-edges between them. Instead of merge, we use the word contract.

Exercise 6 *For the example shown in the attached slides, calculate the probability of that particular sequence being taken by the algorithm.*

Exercise 7 *If the minimum cut size is k, show that the minimum degree of G is at least k, and that hence the number of edges is at least $nk/2$.*

We shall show in the second class that Karger's algorithm outputs a given minimum cut with probability at least $\frac{2}{n(n-1)}$. The probability of finding a minimum cut can be increased by repeating the algorithm several times, and taking the minimum cut among all the cuts that were output.

2.2 Probability Amplification

Let E be an experiment that we perform, and which has a probablity p of success. If we repeat the experiment k times independently, then the probability of failing every time is equal to $(1-k)^p$. In particular, if we repeat the experiment $\frac{c}{p}$ times, then the probability of failure is at most e^{-c} (where we used the fact that $e^x \geq 1+x$). Taking a value of $c = 10$ for example, gives a probability of success more than 99.9%.

3 Review

3.1 Probability amplification

Suppose that we have an algorithm A with success/correctness probability p. If A is repeated $\frac{c}{p}$ times, then the probability that A fails every time is $(1-p)^{m/p} \leq e^{-c}$.

Here, we used the fact that $e^{-x} \geq 1-x$. $\log(1-x) = -x - x^2/2 - x^3/3 - \ldots \leq -x$.

3.2 Exercises: discussion

1. The polynomials are different: at $x = 2$, or modulo 3.

   Ex: Verify if $AB = C$ for $n \times n$ matrices in $O(n^2)$ time.

2. Decision problem: $> \frac{1}{2}$, Search problem: $> \frac{1}{poly(n)}$.

3. Simulating a real coin: let I be the set of indices at which the binary representation of p has a 1. Then the probability that the first heads occurs in an index in I is p.

4. Simulating a fair coin using a biased coin: end-of-class; hint: HT and TH have the same probability.

3.3 Min-cut

Review the algo. Number of contractions?

The probability in the exercise is: $\frac{1}{12} \times \frac{1}{11} \times \frac{2}{10} \times \frac{1}{8} \times \frac{3}{7} \times \frac{2}{4}$.

The minimum cut size is at most the size of a cut that separates v from $V \setminus \{v\}$. So $deg(v) \geq k$ for all v. Adding over all vertices, we get $2m \geq nk$.

Let C be a fixed minimum cut. There can be more than one; for example, K_n has n minimum cuts. Let E_i be the event that none of the edges of C are merged during the ith round.

We have:

$$Pr[E_1 \wedge E_2 \wedge E_n] = Pr(E_1)Pr(E_2|E_1)Pr(E_3|E_2 \wedge E_1) \ldots Pr(E_n|E_1 \wedge \ldots \wedge E_{n-1}).$$

The probability of the edges of C surviving the ith round, given that they have survived so far is $Pr[E_i|E_1 \wedge \ldots \wedge E_{i-1}] = \sum_r Pr[m_i = r]\frac{(m_i - k)}{m_i}$, where m_i is the number of edges at the beginning of the ith round. We simply note that the number of vertices at this stage is $n + 1 - i$ and hence $m_i \geq (n + 1 - i)k/2$.

Thus, $Pr[E_1 \wedge E_2 \ldots E_{n-1}] \geq \prod_{i=1}^{n-2} \left(1 - \frac{2}{n+1-i}\right) = \frac{2}{n(n-1)}$.

Exercise 1 Calculate the probability for K_4 with a fixed min cut.

4 Random variables, Expectation, and Quick-Sort

4.1 Random variables

Example 1: List of names, and first letter. Has its own distribution. Function from Ω to any.

Example 2: X is a r.v. uniform in $[0, 1]$.

Real-valued random variables.

A bet: I think of a 3-digit number. If your prediction is correct all times, I pay $Rs.100$, else you pay $Rs.100$. How about: I pay $Rs.100000$, else you pay $Rs.10$.

X is a r.v. taking values from a list $A_1, A_2, \ldots$ i.e. $\Omega(X)$ is discrete.

Let $p(X = A_i) = p_i$; then $\sum_{i=1} p_i = 1$.

Probability mass function:

$$f(x) = Pr[X = x].$$

Example 1: Let X take values 1,2,3,4 with probabilities as below.

x	1	2	3	4
f(x)	1/3	1/4	1/6	1/4

Find $E[X]$, $E[X^2]$, $E[1/X]$.

Example 2: Toss a die. Expected value. Toss a pair of dice. Expected value of the sum?

Linearity of expectation; expected number of heads, expected number of successes.

Example 3: 50 students and 50 seats. Go out and come back and sit in a random seat. What is the expected number of students who sit in their original seat?

Exercise: Number of distinct elements

Define uniform distribution; geometric distribution and biased coin tosses; expected time to succeed.

Exercise: Bacteria replication

Independence

Definition.

Example 1: Two tosses

Example 2: A: Both heads, B: At least one head

Example 3: Toss two dice. A: X+Y is even, B: X is even, C: X-Y is even.

Exercise 2: Find three events A, B, C such that A, B and A, C are dependent, and B, C are independent.

4.2 Expectation

Definition: Discrete and Continuous, density function and find c if cx^2 is a pdf in $[0, 1]$.

Example 1: One die

Example 2: Uniform in $\{1, 2, \ldots, n\}$ and in $[a, b]$, and for above example.

Example 3: Bet of 6-digit number

Exercise 3: Expected number of trials for success (geometric distribution)

Linearity of expectation: 'Two dice

Number of heads: Bernoulli and Binomial distribution **Exercise 4:** Expected number of fixed points. Expected running time of bubble sort? Number of inversions

Exercise: Coupon Collector problem

Recurrence Uniform, number of trials

Exercise 5: Expected number of tosses to get consecutive heads

Exercise 6: Probability of bacteria surviving

Conditional expectation: $E[X|A] = \sum_{x \in X} xP(x|A)$.

Toss a die. If small, toss again.

Linear transformation, independence and product.

4.3 Analysis of quick-sort

What is the probability that a_i and a_j are compared? It is the probability that the first element in $a_i, a_{i+1}, \ldots, a_j$ that becomes a pivot, is equal to a_i or a_j. This is $\frac{2}{j-i+1}$.

Thus, the expected number of comparisons is (using linearity of expectation) equal to: $\sum_{i=2}^{n} \frac{n+1-i}{i} = (n+1)(H_n - 1) - (n-1) \sim n \log n$.

5 Markov's Inequality

The running time for quicksort is a random variable with expecation $\sim n \log n$. What is the probability that quicksort actually takes much longer than that, say that it makes n^2 comparisons? Is it the case that a random variable will with high probability not become too much larger (or smaller) than its mean?

Another example: Consider a bet in which your expected payoff is Rs. 1000,000,000. But you have to pay Rs.1,000,000 to enter the bet, and the probability that you win is one in a million. Would you like to take the bet assuming that it's a one-time chance?

As the above example shows, the expected value of a variable may be high while the probability of being much smaller may be very high (say as close to 1 as we like).

The following tool shows that the converse cannot happen for non-negative random variables, i.e. the probability of being much larger than the mean is necessarily small.

Theorem 1 (Markov's Inequality) *If X is a non-negative real-valued r.v. and $t > 0$, then:*

$$Pr[X \geq t] \leq E[X]/t.$$

If you know $X \geq a$ AND μ, then you know upper bounds on $P(X \geq t)$.

If you konw $X \leq b$ AND μ, then you know upper bounds on $P(X \leq t)$.

Only the distance between μ and the bounds that you have matters.

Sharpness: This inequality is sharp: it is tight when X takes the value t with probability μ/t and zero with remaining probability.

Example: Suppose that $X \geq 0$ and we know $E[X] = 40$. Then we can say that $Pr[X \geq 60] \leq 2/3$. Note that we also obtain $Pr[X \leq 60] \geq 1/3$.

On the other hand, we can't give an upper bound on $Pr[X \geq 30]$; firstly, Markov's inequality gives an upper bound of 2, which is useless. Secondly, this probability can be arbitrarily high: let's set $Pr[X = 30] = 0.9999$; now it should be clear that we can still achieve the mean of 40 by choosing a suitable much larger value with the remaining probability.

We also cannot give a lower bound on $Pr[X \geq 60]$ or on $Pr[X \geq 35]$. The first probability can be zero with X taking just the value 40 with probability one. The second probability can be made arbitrarily small, as in the example in the previous paragraph.

Usefulness: It is useful for *upper bounds* on the probability of taking large values, specifically values larger than μ. This is equivalent to having lower bounds on taking small values.

It cannot be directly applied to get upper bounds on the probability of taking small values. Equivalently, it cannot be directly applied to get lower bounds on the probability of taking large values.

In more words, we can say that with high probability, X will be small, but we can never say (using Markov's inequality) with high probability that X will be large.

Improving Markov:

With additional information, can we get (i) a smaller upper bound on $Pr[X \geq t]$? (ii) an upper bound for smaller values of t than μ?

Exercise: Let $X \geq 0$ be a r.v. with $E[X] = 40$. Suppose now that we have the additional information that $E[\sqrt{X}] = 5$ and $E[X^2] = 2000$.

Show that $Pr[X \geq 30] \leq 0.92$ and $Pr[X \geq 60] \leq 5/9 \sim 0.55$.

A key idea to getting more from Markov's inequality is to apply it as follows:

Consider a non-decreasing function $f : \mathbb{R} \to \mathbb{R}^+$. Then we have:

$$Pr[X > t] < E[f(X)]/f(t).$$

To improve upon the pure Markov, we need: $E[f(X)]/f(t) < E[X]/t$, which can be rewritten as:

$$f(t)/t > E[f(X)]/E[X].$$

Example 1: $f(X) = aX+b$. Then we want $a+b/t > a+b/E[X]$ or $t < E[X]$. But in this range, Markov's inequality is not applicable so linear functions alone don't improve Markov.

Example 2: $f(X) = X^2$. For the bound to be better, we want $t > E[X^2]/E[X]$.

6 Chebyshev's Inequality

6.1 Review of Markov's Inequality

Exercise 8 *Let $X \geq 0$, $E[X] = 10$. What can you say about $Pr[X \geq 50]$?*

Let $X \geq -10$ and $E[X] = 10$. What can you say about $Pr[X \geq 50]$?

Let $X \leq 20$ and $E[X] = 10$. What can you say about $Pr[X \leq 0]$?

Concentration around the mean using Markov:

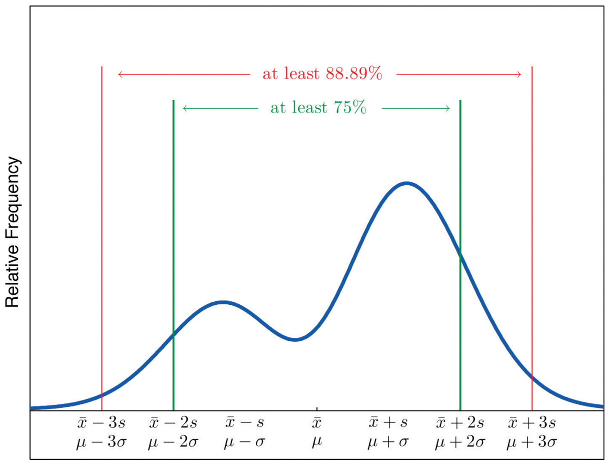
Let $X = X_1 + \ldots + X_n$, where $Pr(X_i = 1) = p$, $Pr(X_i = 0) = 1-p$. Consider the values $p = 1/2$. What can we say about $Pr(X > 0.7n)$?

Using Markov's inequality directly, we can say that $Pr(X > 0.7n) < \frac{0.5n}{0.7n} = 5/7$. This is however a bad bound, and it is possible to use symmetry to improve this to 5/14, which is still not a good bound for large values of n. We will improve this a lot in the next section by using the variance of X.

Theorem 2 *Let X be a random variable with expectation μ and standard deviation σ. Then we have:*

$$Pr[|X - \mu| \geq k\sigma \leq \frac{1}{k^2}.$$

For example, $Pr[|X - \mu| \geq 5\sigma] \leq \frac{1}{25}$, and $Pr[|X - \mu| \geq 10\sigma] \leq \frac{1}{100}$.



Exercise 9 *Suppose that the lifetime of light-bulbs from a shop is distributed randomly with a mean of 200 hours. If the variance is known to be at most 100, show that with probability at least 96%, the bulb will last for at least 150 hours.*

Variance of Binom(n,p):

Let X be the sum of n identical Bernoulli variables $X_1, \ldots, X_n$ with probability p. Then, we have $Var[X] = nVar[X_i] = n(p - p^2)$.

In particular, when $p = 1/2$, we get $Var[X] = \frac{n}{4}$ and $\sigma[X] = \frac{\sqrt{n}}{2}$. We can now apply Chebyshev's inequality to obtain:

$$Pr(X > 0.7n) \leq Pr(|X - 0.5n| > 0.2n) \leq \frac{1}{0.16n}$$

where we used $0.2n = 0.4\sqrt{n}\sigma[X]$.

7 Median-Find Problem

Input: $A = a_1, a_2, \ldots, a_n$

Output: $b_{n/2}$, where $b_1 < b_2 < \ldots < b_n$ is the sorted order of the a_is.

We now describe a randomized algorithm. We use parameters $r \sim n^{3/4}$, $k \sim n^{1/8}$, $s = \frac{k\sqrt{r}}{2} \sim \frac{n^{1/2}}{2}$, and $t = \frac{4sn}{r} \sim 2n^{3/4}$.

Algorithm:

1. *Pick $X_1, \ldots, X_r$ independently and u.a.r. from the a_is.*
2. *Sort the X_is to obtain $Y_1, \ldots, Y_r$.*
3. *Set $l = Y_{r/2-s}$ and $u = Y_{r/2+s}$.*
4. *Compare every element with l, u.*
5. *Let $L = \{a \in A : a < l\}, U = \{a \in A : a > u\}$.*

   If $|L| > n/2$ or $|U| > n/2$, ABORT.
6. $M = \{a \in A : l \le a \le u\}$.

   If $|M| > t$, ABORT.
7. *Sort M and return the element whose rank in M is $n/2 - |L|$.*

Analysis of running time:

If the algorithm does not fail, it returns the correct answer in Step 7. Step 4 takes $2n$ comparisons, and the remaining steps together take time $O(n^{3/4} \log n)$, thus the total time is $2n + o(n)$.

Analysis of failure probability:

There are two steps in which the algorithm can fail: in step 5, it may happen that the median lies outside the interval $[l, u]$, or in step 6, it may happen that $|M|$ is too large. We will now bound the probabilities of these two "bad events".

Let E_1 be the event that the l is larger than the median, that is: $Y_{r/2-s} > b_{n/2}$; and let E_2 be the event that $|M \cap \{b_1, \ldots, b_{n/2}\}| > t/2$.

Proposition 3 *For* $s = k\frac{\sqrt{r}}{2}$ *and* $t = \frac{4sn}{r}$, *we have:*

$Prob(E_1) \leq \frac{1}{k^2}$, *and* $Prob(E_2) \leq \frac{1}{k^2}$.

Proof:

The first type of bad event:

The event $E_1 : Y_{r/2-s} > b_{n/2}$ is equivalent to: $|X \cap L| \leq r/2 - s$, where $L = \{b_1, \ldots, b_{n/2}\}$.

We can write $|X \cap L|$ as a sum of indicator variables. Let $Y_i = 1$ if $X_i \in L$ and $Y_i = 0$ otherwise. Thus, $|R \cap L| = \sum_{i=1}^{r} Y_i$; we shall denote this sum by Y for convenience.

We have $E[Y] = \frac{r}{2}$ and $\sigma(Y) = \sqrt{r}/2$.

Thus, $Pr(E_1) = Pr(|X \cap L| \leq r/2 - s) = Pr(Y \leq r/2 - k\sqrt{r}/2) \leq \frac{1}{k^2}$.

The second type of bad event:

The event $|M \cap \{b_1, \ldots, b_{n/2}\}| > t/2$ is equivalent to: $|X \cap T| \geq r/2 - s$, where $T = \{b_1, b_2, \ldots, b_{n/2-t/2}\}$.

Let $Z = |X \cap T|$ so that $E[Z] = \frac{r}{n}(n/2 - t/2) = \frac{r}{2} - \frac{rt}{2n} = r/2 - 2s$ and we know that $\sigma[Z] \leq \sqrt{r}/2$.

Thus, $Pr(E_2) = Pr(Z \geq r/2 - s) \leq Pr(|Z - E[Z]| \leq s) \leq \frac{1}{k^2}$. This completes the proof of Proposition 3.

Finally, we find that the probability that the algorithm fails is at most $2Pr(E_1 \cup E_2) \leq \frac{4}{k^2} = \frac{4}{n^{1/4}}$.

8 Set Balancing

Consider a set $S \subseteq \{0,1\}^m$ with n elements. Think of each element in S as a data-point and each of the m co-ordinates as representing a feature. Our goal is to partition S into two sets S_{red} and S_{blue} (which we can think of as training and test set, for example), so that for every feature i, the elements that have that feature (i.e. having $x_i = 1$) are split roughly equally into S_{red} and S_{blue}.

To model this precisely, let $S_i = \{x \in S : x_i = 1\}$. For a coloring $c : S \to \{Red, Blue\}$ and a subset $T \subseteq S$, we define the **discrepancy** of the set T (with respect to the coloring) as $disc(T, c) = ||\{x \in T | c(x) = Red\}| - |\{x \in T | c(x) = Blue\}||$.

We then want to find a coloring c such that $disc(T, c)$ is small for every set T in $\{S_1, S_2, \ldots, S_n\}$. Using the Chernoff bound, we show that a random coloring is pretty good.

8.1 Bounds for a random coloring

Proposition 4 *Consider a random coloring c which colors every element in S independently with a color u.a.r from $\{Red, Blue\}$. Then with probability at least $1 - \frac{2}{m}$, we have for every i: $disc(S_i, c) \leq 2\sqrt{n \log m}$.*

Proof Consider a set T from $\{S_1, S_2, \ldots, S_m\}$ (recall that S_i consists of those elements which have ith bit equal to one). For each element $x \in T$, let $Z(x) = +1$ if $c(x) = Red$ and $Z(x) = -1$ if $c(x) = Blue$.

Then $disc(T, c) = |Z(T)|$, where $Z(T) = \sum_{x \in T} Z(x)$.

We note that $E[Z(T)] = 0$ because $E[Z(x)] = 0$ for every x, and since each $Z(x)$ takes values in $[-1, 1]$, we can apply a Chernoff bound to get: for $t > 0$,

$$Prob(disc(T, c) \geq t) \leq 2e^{\frac{-t^2}{2k}}.$$

Here k is the number of random variables in the sum, which is $|T|$ and hence at most n.

Thus, we have: for every T and every $t > 0$,

$$Prob[disc(T,c) \geq t] \leq 2e^{\frac{-t^2}{2n}}.$$

Now substituting $t^2 = 4n \log m$, we get $Pr[disc(T,c) \geq t] \leq \dfrac{2}{m^2}$.

Finally, by the union bound, the probability that there is some T with a discrepancy larger than t is at most $n\dfrac{2}{m^2} = \dfrac{2}{m}$. Thus, with probability at least $1 - 2/m$, all the sets have discrepancy bounded by $2\sqrt{n \log m}$. ■

9 The Hoeffding Bound

Theorem 5 *Let $\{X_i\}_{i=1}^n$ be a set of independent random variables in $[0,1]$ and $X = \sum_{i=1}^n X_i$ with $E[X] = \mu$. Then, for every $t > 0$, we have:*

$$Pr[|X - \mu| > t] < 2e^{-2t^2/n}.$$

Equivalently, $Pr[|X/n - \mu/n| > t] < 2e^{-2nt^2}$.

9.1 Hoeffding's Lemma

Lemma 6 *Let X be a r.v. in $[0,1]$ with expectation μ, and $\alpha \geq 1$. Then $E[\alpha^X] \leq (\alpha - 1)\mu + 1 \leq e^{(\alpha-1)\mu}$.*

Proof α^x is convex; thus in the interval $[0,1]$, the curve $y = \alpha^x$ lies below the line joining $(0,1)$ and $(1,\alpha)$. This line has the equation $y = (\alpha-1)x+1$; thus $\alpha^x \leq (\alpha-1)x + 1$ in this interval. Taking expectations on both sides gives the desired result. □

Lemma 7 *[Hoeffding's Lemma] Let X be a r.v. in $[a,b]$ with mean μ and let $L = b - a$. Then, for $\alpha \geq 1$, we have:*

$$E[\alpha^X] \leq \alpha^{\mu} \frac{(b\alpha^a - a\alpha^b)}{(b-a)} \leq \alpha^{\mu + (\log\alpha)L^2/8}.$$

We skip the proof of Lemma 7, but as can be guessed, the first inequality is again obtained from the convexity of α^X; the second inequality involves writing $\alpha = e^{\log \alpha}$, expanding the Taylor series on both sides, and comparing coefficients; this can be painful to do and is therefore left as an optional exercise.

Proof Proof of Theorem 5 We prove one half of the inequality.

We have: $Pr[X > \mu + t] = Pr[\alpha^X > \alpha^{\mu+t}] < \dfrac{E[\alpha^X]}{\alpha^{\mu+t}} = \alpha^{-\mu-t} \prod_{i=1}^n E[\alpha^{X_i}] \leq \alpha^{-\mu-t}\alpha^{\mu+n(\log\alpha)/8}$, where the last inequality is obtained by applying Hoeffding's Lemma to each of the X_is.

Now we set $\alpha = e^{4t/n}$ so that $\alpha^t = e^{4t^2/n}$ and $\alpha^{n(\log\alpha)/8} = e^{2t^2/n}$. $\square$

The same calculation in the previous proof can be repeated to obtain the following.

Theorem 8 *Let $\{X_i\}_{i=1}^n$ be a set of independent random variables, with $X_i \in [a_i, b_i]$ and $X = \sum_{i=1}^n X_i$ with $E[X] = \mu$. Set $L = \sum_{i=1}^n (b_i - a_i)^2$. Then, for every $t > 0$, we have:*

$$Pr[|X - \mu| > t] < 2e^{-2t^2/L}.$$

In the proof of Theorem 5, we used Lemma 2 to upper-bound $E[\alpha^{X_i}]$. By using Lemma 1 instead, we can obtain the following, whose proof is skipped.

Theorem 9 *If X is the sum of independent random variables in $[0,1]$, and $t \geq 6E[X]$, then:*

$$Pr[X > t] < 2^{-t}.$$

Note that the bound in Theorem 5 is independent of the number of summands, so this result is useful when the number of summands is much larger than $E[X]$ or t.

Exercise: Obtain an exponentially small upper bound on the probability that a fair coin tossed n times results in Heads more than $0.6n$ times.

9.2 Application of the Hoeffding Bound to boosting decision probability

Let A be an algorithm that can solve a decision probability with probability $\frac{1}{2} + \alpha$ where $\alpha > 0$. Suppose that we repeat A $2k+1$ times and take the majority vote of the algorithm's answer. How should we choose k so that the probability of the majority decision being incorrect is at most ε?

Let $X_i = 1$ if the algorithm's ith answer is correct, and let $X_i = 0$ otherwise. Let $X = X_1 + \ldots + X_{2k+1}$. Then $E[X] = (2k+1)(1/2+\alpha) > k + 2k\alpha$.

$Pr(\text{Majority is incorrect}) = Pr(X \leq k) < Pr(|X - E[X]| \geq 2k\alpha)$. By the first Chernoff bound, this probability is less than $e^{-4k^2\alpha^2/(2k+1)}$. Thus, we require: $e^{-4k^2\alpha^2/(2k+1)} < \varepsilon$, which is equivalent to: $4k^2/(2k+1) > \log(1/\varepsilon)\frac{1}{\alpha^2}$.

This tells us that for fixed α, the number of times we need to repeat A to reduce the error probability to ε is of the order $O(\log(1/\varepsilon))$; in contrast, an analysis using Chebyshev's inequality gives a much poorer bound.

9.3 Some useful inequalities

1. $(1-x) \leq e^{-x}$ for all x.
2. $(1-x) \geq 4^{-x}$ for $0 \leq x \leq 1/2$.
3. $(n/e)^n \leq n! \leq en(n/e)^n$.
4. $(n/k)^k \leq \binom{n}{k} \leq (en/k)^k$.

10 Network Routing

The Problem: We are given N pairs of source and destination pairs (each of the N vertices being a source as well as a destination) in a Hamming network with N nodes, and our goal is to design a routing algorithm so that all the packets are transmitted efficiently, specifically within $O(\log_2 N)$ steps.

Before we understand the problem statement, let's first consider the design of a good communication network. Some of the desirable properties are sparsity (fewer number of connections), high connectivity (remains connected even when a few nodes fail), and low diameter (short paths b/w any pair of nodes).

A simple candidate is the Hamming network, whose vertex set is $V = \{0,1\}^n$ and the edge-set is $E = \{(x,y) | H(x,y) = 1\}$. Here, $H(x,y)$ is the Hamming distance which is the number of positions i in which $x_i \neq y_i$.

The Hamming network is n-regular, and has connectivity and diameter each equal to n. We denote the number of vertices by N so that $n = \log_2 N$.

10.1 Packet transmission and the bit-fixing algorithm

A packet can be transmitted from a source node to a destination node, where it follows a designated path. We assume that traversing each edge takes the same time, and thus the number of time-steps is measured as the number of edges on the path.

Transmitting a single packet from a node to a destination in the Hamming network thus takes at most $n = \log_2 N$ steps. There are several shortest paths between each pair of nodes, and the bit-fixing algorithm takes the path from x to y in which the bits of x that are different from y are flipped in order. For example, the bit-fixing path from 01010 to 11001 is 01010 $\rightarrow$ 11010 $\rightarrow$ 11000 $\rightarrow$ 11001.

Parallel routing:

It is possible to transmit several packets between various nodes in the network simultaneously. However parallel transmission can happen only across different edges. For example, if several packets arrive at the same node at the same time, and wish to traverse the same edge, then they must do so one after another. The waiting packets put in a buffer at that node and for each edge out of that node, some priority queue is maintained to decide the order in which the packets will be sent along the edge.

Now we restate the probem, where the routing algorithm is understood to use parallel routing.

The Problem: We are given N pairs of source and destination pairs (each of the N vertices being a source as well as a destination), and our goal is to design a routing algorithm so that all the packets are transmitted within $O(\log_2 N)$ steps.

The bit-fixing algorithm doesn't work: The bit-fixing algorithm can however take as long as $\Omega(\sqrt{n})$ steps for some inputs. An example of a bad input is: for each source x, write $x = x_L x_R$ where x_L is the first $n/2$ bits of x, and let the destination be $x_R x_L$. The bit-fixing algorithm traverses via $x_R x_R$ and thus all packets whose source string has the same right-half (i.e. $2^{n/2}$ packets) reach a common node. This means that at least $\frac{1}{n}2^{n/2}$ of these packets must traverse the same edge, and hence the algorithm takes time at least $\frac{1}{n}2^{n/2}$.

Finally, we remark that fixing the bits in random order can still lead to high congestion at a node and take super-polynomia (in n) steps.

10.2 Algorithm: Random intermediate node

For each source-destination pair (x, y), the source x picks a random intermediate node z, and sends its packet to z using the bit-fixing algorithm, and then the packet travels from z to the original destination y, again using the bit-fixing algorithm.

Proposition 10 *Using the above algorithm, with probability at least* $1 - \frac{n}{2^{5n}}$, *all packets reach their intermediate node in at most* $7n$ *steps and hence their original destinations in at most* $14n$ *steps.*

To analyze the above algorithm, we use the following lemma from the notes of Tarjan.

Lemma 11 *Suppose that at most* k *packets intersect the route of a packet* P. *Then* P *reaches its destination within* $n + k$ *steps.*

The above claim uses the fact that after two routes overlap and diverge, they will never overlap again. It also uses a clever identity-switching argument, which we do not reproduce here. The intuitive idea is that on average, each packet that intersects the route of P causes a delay of at most one time-step.

Lemma 12 *For a packet P, let E_P be the event that at most $6n$ packets intersect the route of P. Then with probability at least $1 - \frac{n}{2^{6n}}$, E_P holds for all packets P.*

Proposition 1 follows from Lemma 2 and Lemma 3. We now prove Lemma 3 using the following two claims.

Claim 1: If (x, z) and (x', z') are two source-destination pairs whose path traverses the same edge when flipping the ith bit. Then the first i bits of z and z' are identical, and the bits of x and x' from i onwards are identical.

Proof of Claim 1: After flipping the ith bit, both the routes will retain the same first i bits from that point till they reach the destination, thus both destinations must have the same bits in the first i bits. For the second part, observe that both routes have not flipped any of the bits from i onwards when they arrive at the common node, so they must have the same suffix from positions i onwards.

Claim 2: For a given packet P traversing from x to z in the first phase, the expected number of packets that intersect its path is at most $n/2$.

To prove Claim 2, we first fix a source x and let $S_i = \{x' |$x & x' agree on positions i onwards$\}$. For $x' \in S_i$, let $z_i(x') = 1$ if x and x' arrive at the same node when flipping the ith bit. Let Z_i be the number of packets that traverse the same edge as the packet from x when flipping the ith bit. Thus, $Z_i = \sum_{x' \in S_i} z_i(x')$.

We have: $Pr[z_i(x') = 1] = \frac{1}{2^i}$ because this is the probability that the random intermediate nodes of x and x' share the same bits in the first i positions.

Also, $|S_i| = 2^{i-1}$ and hence $E[Z_i] = \frac{1}{2}$.

Thus, the expected number of packets that share some edge with the route of x is equal to $n/2$, proving Claim 2, and this expectation can be written as the sum of N independent random variables.

Thus, to obain Lemma 3, we apply the second Chernoff bound from the previous bound (restated below), which does not depend on the number of random variables.

Lemma 13 (Chernoff Bound) *If X is a sum of independent random variables in $[0, 1]$, and $t \geq 6E[X]$, then $Pr[X > t] < 2^{-t}$.*