

Lecture 6: Introduction to Randomized Algorithms

1 Probability Review

Definition 1.0.1 *A random variable, X , is a real number that is the outcome of a random event.*

For example, X = number of spots when a six-sided die is rolled.

Definition 1.0.2 *Expectation of X , $E[X] = \sum_i i \cdot (Pr[X = i])$*

Continuing with the six-sided die example, the expectation for a six-sided die roll: $E[X] = \sum_{i=1}^6 i \cdot (Pr[X = i]) = \frac{7}{2}$

Theorem 1.0.3 (Linearity of Expectations) *If X and Y are any two random variables then $E[X + Y] = E[X] + E[Y]$. Generally: $E[\sum X_i] = \sum E[X_i]$.*

If c is a real number then $E[c \cdot X] = cE[X]$. $E[XY] \neq E[X]E[Y]$ unless X & Y are independent.

Definition 1.0.4 (Indicator Random Variables) *Let w be some event (e.g. a six sided die was rolled and turned up 6), the indicator random variable for w :*

$$I_w = \begin{cases} 1 & \text{if } w \text{ happens} \\ 0 & \text{otherwise} \end{cases}$$

Theorem 1.0.5 (Markov's Inequality) $Pr[|X| \geq a] \leq \frac{E[x]}{a}$

Proof 1.0.6

$$\text{Let } I_{|x| \geq a} = \begin{cases} 1 & \text{if } |x| \geq a \\ 0 & \text{if } 0 < |x| < a \end{cases}$$

It is always the case that $\frac{x}{a} \geq I_{|x| \geq a}$

Therefore: $E[\frac{x}{a}] \geq E[I_{|x| \geq a}]$

and by 1.0.3 $\frac{E[x]}{a} \geq Pr[|x| \geq a]$

Definition 1.0.7 (Variance)

$$\begin{aligned} \text{Var}[X] &= E[X - E[X]]^2 \\ &= E[X^2 - 2XE[X] + (E[X])^2 + 2E[X^2]] \\ &= E[X^2] - E[X]^2 \end{aligned}$$

Theorem 1.0.8 (Chebyshev's inequality) $Pr[X - E[X] \geq a] \leq \frac{\text{Var}[X]}{a^2}$

The proof of Chebyshev's inequality from Markov's inequality is left as an exercise for the reader.

2 A Toy Example

Suppose you are given two containers, one containing N blue balls and the other containing $\frac{N}{2}$ blue balls and $\frac{N}{2}$ yellow balls. The goal is to determine which container is which.

2.1 Algorithm 1

1. Pick a container at random
2. Draw 20 balls
3. If a yellow ball is found, stop and report "found the mixed container"

4. Else stop and report “found the blue container”

This algorithm is fast, but not always correct. However, it has a high probability of being correct.

2.2 Algorithm 2

1. Pick a container at random and draw a single ball
2. If a yellow ball is found, stop and report “found the mixed container”
3. Else goto step 1

This algorithm is always correct, but not always fast. However, it has a high probability of being fast.

2.3 Algorithm 3

1. Pick a container at random and start drawing balls
2. If a yellow ball is found, stop and report “found the mixed container”
3. Else continue until the container is empty and report “found the blue container”

This algorithm is always correct, but with $P[\frac{1}{2}]$ it is slow.

3 Probabilistic Algorithms

There are two classes of probabilistic algorithms we will discuss. Monte Carlo algorithms are always fast, not always correct, but correct with high probability. Las Vegas algorithms are always correct, not always fast, but fast with high probability. A formal definition follows.

Definition 3.0.1 (Monte Carlo Algorithm) A Monte Carlo algorithm is a probabilistic algorithm M for which \exists a polynomial P such that $\forall x$, M terminates within $P(|x|)$ steps on input x .

Furthermore, $P[M(x) \text{ is correct}] > \frac{2}{3}$ where probability is taken over all coin tosses of algorithm M

Definition 3.0.2 (Las Vegas Algorithm) A Las Vegas algorithm is a probabilistic algorithm M for which \exists a polynomial P such that $\forall x$, $E[\text{running time}] = \sum_{t=1}^{\infty} (t)Pr[M(x) \text{ takes exactly } t \text{ steps}] < P[x]$.

Furthermore, the output of M is always correct.

4 The Max-Cut Problem

Given a graph, $G = (V, E)$, we wish to partition V into two sets, A and B , such that the number of edges crossing the cut is maximized. This problem is NP-Hard.

4.1 The Erdős and Spencer Probabilistic Method

We now use the Probabilistic method to show that for any graph, there is guaranteed to exist a cut that contains at least half the edges of the graph. Notice that such a cut is always a $\frac{1}{2}$ approximation to max cut (since no cut can contain more than all the edges of a graph!) Then we detour through a randomized algorithm to show a deterministic $\frac{1}{2}$ approximation of max-cut.

For each vertex, i , flip a coin: -1=tails, 1=heads. For $x_i = -1$, put i in A . For $x_i = 1$, put i in B . $E[x_i] = 0$.

Define $E_{ij} = 1 - \frac{x_i x_j}{2} \forall i, j \in V$ (1 if crosses cut, 0 if not)

Let $S =$ Number of edges that cross the cut.

$$\begin{aligned}
 E[S] &= E \left[\sum_{(i,j) \in E} E_{ij} \right] \\
 &= \sum_{(i,j) \in E} E[E_{ij}] \text{ by Linearity of Expectation} \\
 &= \sum_{(i,j) \in E} E \left[\frac{1 - x_i x_j}{2} \right] \\
 &= \sum E \left[\frac{1}{2} \right] - E \left[\frac{x_i x_j}{2} \right] \\
 &= E[x_i]E[x_j] \text{ by independence of these variables}
 \end{aligned}$$

A proof that this algorithm gives $\geq \frac{E}{4}$ edges with $Pr > \frac{1}{2}$ is left as an exercise to the reader.

This probabilistic argument says that if $Exp \geq \frac{|E|}{2}$, then there must exist some assignment of vertices to sets, A & B , that is at least this good.

A Digression on Pairwise Independence

x_i	x_j	x_k
1	1	0
1	0	1
0	1	1
0	0	0

In the table above, any one variable can be removed leaving the remaining two to be pairwise independent.

In the context of a probabilistic algorithm, this is a reduced set of coins that will suffice.

$$E[X_s = a | x_t = b] = E[x_s = a] \text{ where } s \neq t \in \{i, j, k\}$$

With N variables, and $x_i \in \{-1, 1\}$, we wish to construct a small sample

space such that each $x_i x_j$ is pairwise independent. We assume that $N = 2^k$ for some integer k (if not, just add extra variables to get to the next largest power of 2). We represent w as $w_1 w_2 w_3 \dots w_{\log_2 N}$, a $\log N$ bit sequence of 0's and 1's.

Let $x_i = (-1)^{\text{bin}(i) \odot w}$, where $\text{bin}(i)$ is the binary expansion of i and $x \odot y$ is the count of bits set in the bit-wise exclusive or of x and y .

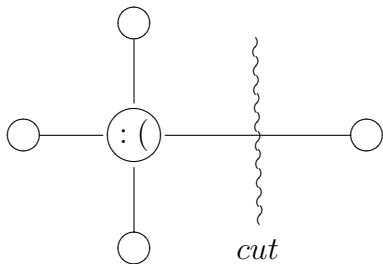
For example, $1101\ 0010 \odot 0001\ 1011 = 4$

Notice x_i will be 1 if this number is even and -1 if this number is odd.

To show: for all $i \neq j$, $Pr[x_i = 1 | x_j = 1] = 1/2$ and $Pr[x_i = -1 | x_j = 1] = 1/2$. Therefore, all variables in the sample space are pairwise independent, and the sample space is $2^{|w|} = O(2^{\log N}) = O(N)$ in size. We can simply check all sample points in polynomial time.

4.2 A Deterministic Algorithm

A vertex can be labeled as “happy” or “unhappy”. Vertices are unhappy if more they have more direct neighbors on their side of the cut than on the other side, and are happy otherwise. Any unhappy vertex can be made happy by flipping it to the other side of the cut.



The greedy algorithm is to simply flip the side of unhappy vertices. That is, while there exists an unhappy vertex, pick an unhappy vertex and flip it across the cut.

While a flip may cause some neighbors to become unhappy, each flip is guaranteed to increase the number of edges that cross the cut, thus forward progress is made with each step and the algorithm will terminate.