Lecture 21: Randomness in Algorithms

built on 2019/07/01 at 22:35:50

The main theme of this lecture is *randomized algorithms*. These are algorithms that make use of randomness in their computation. We will begin this lecture by reviewing concepts that you have seen from Discrete Math—and start to develop machinery needed to analyze algorithms that make use of randomness. By the end of this lecture, you will be well-equipped to design and analyze a randomized algorithm that finds k smallest elements in a sequence. We will show that the algorithm runs in expected $O(n)$ time.

1 Discrete Probability: Let's Flip Some Coins

You probably have heard about probability already. We will spend some time in this lecture reviewing basic probability concepts. To begin, let's suppose we have a *fair* coin, meaning that it is equally likely to land on either side. If we flip this coin, what is the chance that it is going to turn up heads? The answer, as we know already, is

$$\frac{\text{\# of outcomes that meet the condition}}{\text{\# of total outcomes}} = \frac{1}{2}$$

because in this case, these outcomes are equally likely to happen.

In a more formal setting, we have a *probability space*, which is made up of the following 3 components: First, we have a set of possible outcomes, known as a *sample space* Ω. Second, we have a family of sets $\mathcal{F}$ of allowable events, where each set in $\mathcal{F}$ is a subset of the sample space Ω. Third, we have a probability function $\mathbf{Pr} : \mathcal{F} \to [0, 1]$ satisfying

1. $\mathbf{Pr}[\Omega] = 1$; and
2. for any finite (or countably infinite) sequence of disjoint events $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$,

$$\mathbf{Pr}\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] = \sum_{i=1}^{n} \mathbf{Pr}[\mathcal{E}_i]\,.$$

In the previous example, the sample space is $\{H, T\}$, denoting heads and tails respectively. The family of sets of allowable events $\mathcal{F}$ is simply $\{\{H\}, \{T\}\}$ because that coin has to come up either heads or tails, but not both at the same time.

Often, each outcome in the sample space is equally likely to happen. In this case, the probability of an event $\mathcal{E} \in \mathcal{F}$ can be found by counting the number of outcomes in the event and dividing by the total number of possible outcomes; that is,

$$\mathbf{Pr}[\mathcal{E}] = \frac{|\mathcal{E}|}{|\Omega|}.$$

Independence. Two events A and B are independent if and only if $\mathbf{Pr}[A \cap B] = \mathbf{Pr}[A] \cdot \mathbf{Pr}[B]$.

1.1 Random Variables and Expectation

A *random variable* is a function $f : \Omega \to \mathbb{R}$. We typically denote random variables by capital letters $X, Y, \ldots$. That is, a random variable assigns a numerical value to an outcome $\omega \in \Omega$. For example, we could have a random variable X which will be 1 if the coin turns up heads and 0 otherwise. This particular type of random variables is called indicator random variables: for an event $\mathcal{E}$, the *indicator random variable* $\mathbb{I}\{\mathcal{E}\}$ takes on the value 1 if $\mathcal{E}$ occurs and 0 otherwise.

For a (discrete) random variable X and a number $a \in \mathbb{R}$, the event "$X = a$" is the set $\{\omega \in \Omega : X(\omega) = a\}$. Therefore,

$$\mathbf{Pr}[X = a] = \mathbf{Pr}[\{\omega \in \Omega : X(\omega) = a\}]$$

The *expectation* (or expected value) of a random variable is simply the weighted average of the value of the function over all outcomes. The weight is the probability of each outcome, giving

$$\mathbf{E}[X] = \sum_{\omega \in \Omega} X(\omega)\,\mathbf{Pr}[\omega] = \sum_{k} k \cdot \mathbf{Pr}[X = k]$$

in the discrete case. Applying this definition, we have $\mathbf{E}[\mathbb{I}\{\mathcal{E}\}] = \mathbf{Pr}[\mathcal{E}]$.

Example 1.1 *The expectation of the variable X representing the value of a fair die is*

$$\mathbf{E}[X] = \sum_{i=1}^{6} i\,\mathbf{Pr}[X = i] = \sum_{i=1}^{6} i \times \tfrac{1}{6} = \frac{7}{2}.$$

Example 1.2 *A fair coin is flipped* three *times. Let X be the random variable that gives the number of heads. What's the expected value of X?*

$$\begin{aligned}\mathbf{E}[X] &= \frac{1}{8}\Big(X(HHH) + X(HHT) + X(HTH) + X(HTT) \\ &\qquad + X(THH) + X(THT) + X(TTH) + X(TTT)\Big) \\ &= \frac{1}{8}(3 + 2 + 2 + 1 + 2 + 1 + 1 + 0) \\ &= \frac{12}{8} = \frac{3}{2}.\end{aligned}$$

1.2 Linearity of Expectations

One of the most important theorem in probability is *linearity of expectations*. It says that given two random variables X and Y, $\mathbf{E}[X] + \mathbf{E}[Y] = \mathbf{E}[X + Y]$. It is really powerful because the theorem does not require the events to be independent.

In a general form, it says that if we have random variables $X_1, X_2, \ldots, X_n$, then

$$\mathbf{E}[X_1 + X_2 + \cdots + X_n] = \mathbf{E}[X_1] + \mathbf{E}[X_2] + \cdots + \mathbf{E}[X_n].$$

1.3 Examples

To see why this might be use, let's revisit the example above.

Example 1.3 *If a fair coin is flipped* three *times and X is the random variable that gives the number of heads, what's the expected value of X?*

To use linearity of expectation, we could set

$$\begin{aligned}X_1 &= \text{\# of heads from the first flip} \\ X_2 &= \text{\# of heads from the second flip} \\ X_3 &= \text{\# of heads from the third flip.}\end{aligned}$$

The total number of heads is clearly $X = X_1 + X_2 + X_3$. *But we also know that* X_1 *is either* 0 *or* 1, *and*

$$X_1 = \begin{cases} 1 & \text{if it turns up heads} \\ 0 & \text{if it turns up tails} \end{cases}$$

Therefore,

$$\mathbf{E}[X_1] = 1 \cdot \mathbf{Pr}[\textit{turns up heads}] + 0 \cdot \mathbf{Pr}[\textit{turns up heads}] = 1 \times \frac{1}{2} + 0 \times \frac{1}{2} = 1/2.$$

We also know that X_1, X_2, *and* X_3 *behave in the same way, so by linearity of expectations,*

$$\mathbf{E}[X] = \mathbf{E}[X_1 + X_2 + X_3] = \mathbf{E}[X_1] + \mathbf{E}[X_2] + \mathbf{E}[X_3] = \tfrac{1}{2} + \tfrac{1}{2} + \tfrac{1}{2} = \tfrac{3}{2}.$$

Let's explore a more general setting of n coins.

Example 1.4 *Suppose we toss n coins, where each coin has a probability p of coming up heads. What is the expected value of the random variable X denoting the total number of heads?*

Solution I: We'll apply the definition of expectation directly. This will rely on our strength to compute the probability and the simplify the sum:

$$\begin{aligned}
\mathbf{E}[X] &= \sum_{k=0}^{n} k \cdot \mathbf{Pr}[X = k] \\
&= \sum_{k=1}^{n} k \cdot p^k (1-p)^{n-k} \binom{n}{k} \\
&= \sum_{k=1}^{n} k \cdot \frac{n}{k} \binom{n-1}{k-1} p^k (1-p)^{n-k} && \text{[because } \binom{n}{k} = \frac{n}{k}\binom{n-1}{k-1} \text{]} \\
&= n \sum_{k=1}^{n} \binom{n-1}{k-1} p^k (1-p)^{n-k} \\
&= n \sum_{j=0}^{n-1} \binom{n-1}{j} p^{j+1} (1-p)^{n-(j+1)} && \text{[because } k = j+1 \text{]} \\
&= np \sum_{j=0}^{n-1} \binom{n-1}{j} p^{j} (1-p)^{(n-1)-j)} \\
&= np(p + (1-p))^n && \text{[Binomial Theorem]} \\
&= np
\end{aligned}$$

That was pretty tedious :(

Solution II: We'll use linearity of expectations. Let $X_i = \mathbb{I}\{i\text{-th coin turns up heads}\}$. That is, it is 1 if the i-th coin turns up heads and 0 otherwise. Clearly, $X = \sum_{i=1}^{n} X_i$. So then, by linearity of expectations,

$$\mathbf{E}[X] = \mathbf{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbf{E}[X_i]\,.$$

What is the probability that the i-th coin comes up heads? This is exactly p, so $\mathbf{E}[X] = 0 \cdot (1-p) + 1 \cdot p = p$, which means

$$\mathbf{E}[X] = \sum_{i=1}^{n} \mathbf{E}[X_i] = \sum_{i=1}^{n} p = np.$$

Example 1.5 *A coin has a probability p of coming up heads. What is the expected value of Y representing the number of flips until we see a head? (The flip that comes up heads counts too.)*

Solution I: We'll directly apply the definition of expectation:

$$\begin{aligned}
\mathbf{E}[Y] &= \sum_{k \geqslant 1} k(1-p)^{k-1} p \\
&= p \sum_{k=0}^{\infty} (k+1)(1-p)^k \\
&= p \cdot \frac{1}{p^2} && \text{[by Wolfram Alpha, though you should be able to do it.]} \\
&= 1/p
\end{aligned}$$

Solution II: Alternatively, we'll write a recurrence for it. As it turns out, we know that with probability p, we'll get a head and we'll be done—and with probability $1 - p$, we'll get a tail and we'll go back to square one:

$$\mathbf{E}[Y] = p \cdot 1 + (1-p)\Big(1 + \mathbf{E}[Y]\Big) = 1 + (1-p)\,\mathbf{E}[Y] \implies \mathbf{E}[Y] = 1/p.$$

by solving for $\mathbf{E}[Y]$ in the above equation.

2 Random Puzzles

Before we move on to some real applications. Let's pause slightly and think about a few puzzles related to probability.

Q: How many permutation on $[n] := \{1, 2, 3, \ldots, n\}$ are there? *A:* $n!$

Q: If we pick a random permutation on $[n]$, what's the probability that the first element in the permutation is 1? *A:* $\frac{(n-1)!}{n!} = 1/n$.

Puzzle I: Let S be a sequence of n numbers (not necessarily sorted).

- Pick i uniformly at random from $\{1, 2, 3, \ldots, n\}$.
- Consider $A = \{x \in S : x \leqslant A[i]\}$. What is the size of A in expectation?

(Ans: $\frac{n+1}{2}$)

Puzzle II: Let S be a sequence of n numbers (not necessarily sorted).

- Pick i uniformly at random from $\{1, 2, 3, \ldots, n\}$.
- Consider $A = \{x \in S : x \leqslant A[i]\}$. What is the probability that $|A| \leqslant 3n/4$?

(Ans: $3/4$)

Puzzle III: Same set up as above. What's the probability that $n/4 \leqslant |A| \leqslant 3n/4$?

(Ans: $1/2$)

3 Finding The Bottom k

Consider the following problem:

Input: S — a sequence of n numbers (not necessarily sorted)

Output: a sequence of the smallest k numbers (could be in any order).

Note that the linear-time requirement rules out the possibility of sorting the sequence. Here's where the power of randomization gives you a simple algorithm. For simplicity, we'll assume the elements are unique.

We'll take inspiration from quick sort: We'll pick a random pivot p and split the input into **(1)** less than or equal to p and **(2)** greater than p—then, decide what to do with them next.

As a running example, let's fix $k = 5$. Now say we have

`S = [ 32, 1, 91, 43, 5, 22, 14, 29 ]` and the random pivot chosen is $p = 22$. Therefore,

$$lt = [1, 5, 22, 14] \quad \text{and} \quad gt = [32, 91, 43, 29]$$

What should we do? Ans: we have just found the lowest 4 numbers, but we need one more. Upon a closer look, that will have to come from the smallest number in gt.

Say we have the same S but $p = 43$. Then,

$$lt = [32, 1, 43, 5, 22, 14, 29] \quad \text{and} \quad gt = [91]$$

In this case, what should we do? Should we explore *gt* any further? Not really. We know that *lt* contains all that we need. So it suffices to find the smallest k numbers from *lt*.

In code, we have the following:

Algorithm 1: `findSmallestK`(k, S) — find the smallest k numbers in S

---

if $|S| \leqslant k$ **then**
  return S
else
  Pick $p \in S$ uniformly at random // `p = random.choice(S)`
  Compute $L =$`[x for x in S if x <= p]` and $R =$`[x for x in S if x > p]`
  if $|L| \geqslant k$ **then**
    return `findSmallestK`(k, L)
  else
    return $L +$ `findSmallestK`$(k - |L|, R)$

3.1 Analysis

The question is, *how fast is this algorithm?* We'll try to analyze the running time of this algorithm. The challenge is that we have no idea how big L nor R will be. Worse yet, we can't quite tell whether we're recursing on L or R. So the main idea is we're going to have to be a bit pessimistic.

Let's make an observation now:

> Of the two possible ways to recurse, in either case, `findSmallestK` is called with a sequence of length at most $\max(|L|, |R|)$.

So now define $X_n = \max\{|L|, |R|\}$, which is the size of the larger side. Notice that X_n is an upper bound on the size of the side the algorithm actually recurses into. Because everything else can be done in at most $O(n)$, we have the following recurrence:

$$T(n) \leqslant T(X_n) + O(n)$$

Let's look at this recurrence. We're interested in finding out how this algorithm runs. In some sense, this is captured by the expected running time $\mathbf{E}[T(n)]$.

What's the probability that $\mathbf{Pr}\left[X_n \leqslant \frac{3}{4}n\right]$*?* Since $|R| = n - |L|$, $X_n \leqslant \frac{3}{4}n$ if and only if $n/4 < |L| \leqslant 3n/4$. There are $3n/4 - n/4$ values of p that satisfy this condition. As we pick p uniformly at random, this probability is

$$\frac{3n/4 - n/4}{n} = \frac{n/2}{n} = \frac{1}{2}.$$

Notice that given an input sequence of size n, how the algorithm performs in the future is irrespective of what it did in the past. Its cost from that point on only depends on the random choice it makes after that. So, we'll let $\overline{T}(n) = \mathbf{E}[T(n)]$ denote the expected time performed on input of size n.

Now by the definition of expectation, we have

$$\begin{aligned}
\overline{T}(n) &\leqslant \sum_i \mathbf{Pr}[X_n = i] \cdot \overline{T}(i) + c \cdot n \\
&\leqslant \mathbf{Pr}\left[X_n \leqslant \tfrac{3n}{4}\right] \overline{T}(3n/4) + \mathbf{Pr}\left[X_n > \tfrac{3n}{4}\right] \overline{T}(n) + c \cdot n \\
&= \tfrac{1}{2}\overline{T}(3n/4) + \tfrac{1}{2}\overline{T}(n) + c \cdot n \\
&\implies (1 - \tfrac{1}{2})\overline{T}(n) = \tfrac{1}{2}\overline{T}(3n/4) + c\cdot && \text{[collecting similar terms]} \\
&\implies \overline{T}(n) \leqslant \overline{T}(3n/4) + 2c \cdot n. && \text{[multiply by 2]}
\end{aligned}$$

In this derivation, we made use of the fact that with probability $1/2$, the instance size shrinks to at most $3n/4$—and with probability $1/2$, the instance size is still larger than $3n/4$, so we pessimistically upper bound it with n. Note that the real size might be smaller, but we err on the safe side since we don't have a handle on that.

Finally, the recurrence $\overline{T}(n) \leqslant \overline{T}(3n/4) + 2cn$, if you work it out, solves to $\overline{T}(n) \in O(n)$.