Advanced Algorithms

Lecture 15: Expected running times

Announcements

- **Mid-term grades out**
- Course grading: common question
  - <u>not absolute grading</u>

→ Mid-term course survey.

Randomized algorithms

- Saw some examples:
  - Finding "commonly occurring" element in array
  - Testing if two polynomials are identically equal (circuit design, ...)
  - Checking if a bipartite graph has a perfect matching

    (Tutte polynomial)

Randomized algorithms

run for time r,
success prob $= 1 - \left(\frac{2}{3}\right)^r$

- Saw some examples:
  - Finding "commonly occurring" element in array
  - Testing if two polynomials are identically equal
  - Checking if a bipartite graph has a perfect matching

Finding hay in a haystack…

- Trade-off between "success probability" and running time
- Las Vegas algorithms: always succeed, but running time can be large with some probability (no error)

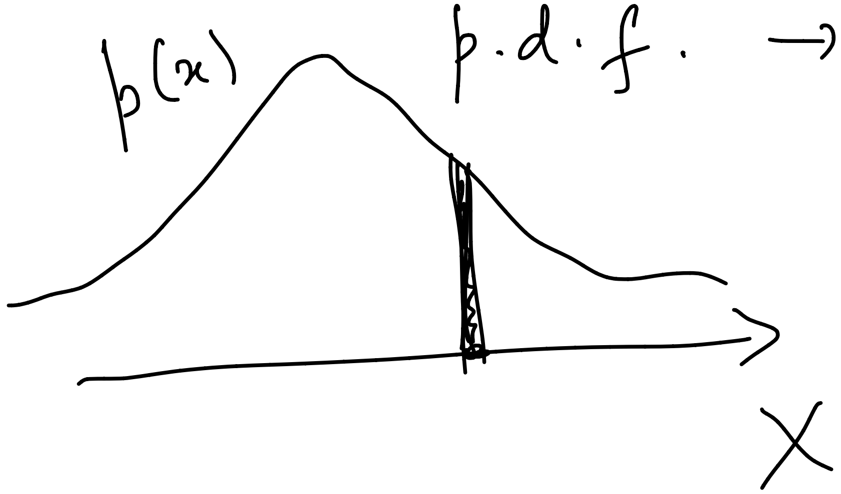
Today’s plan

- Expected running time
- Why expectation is “good enough” (in most cases)

Expectation of a random variable

- <u>Definition:</u> given a random variable X, the expected value of X is

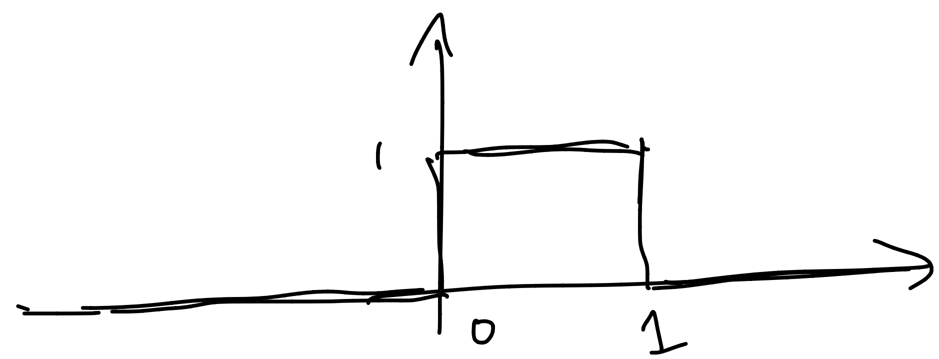
$$\mathbb{E}[X] := \int_{-\infty}^{\infty} x \cdot \underline{p(x)}dx \quad \rightsquigarrow \quad \sum_{x} x \cdot p(X=x)$$



p.d.f. → prob. density function.

$p(x)$ is the p.d.f. of X

Basic examples

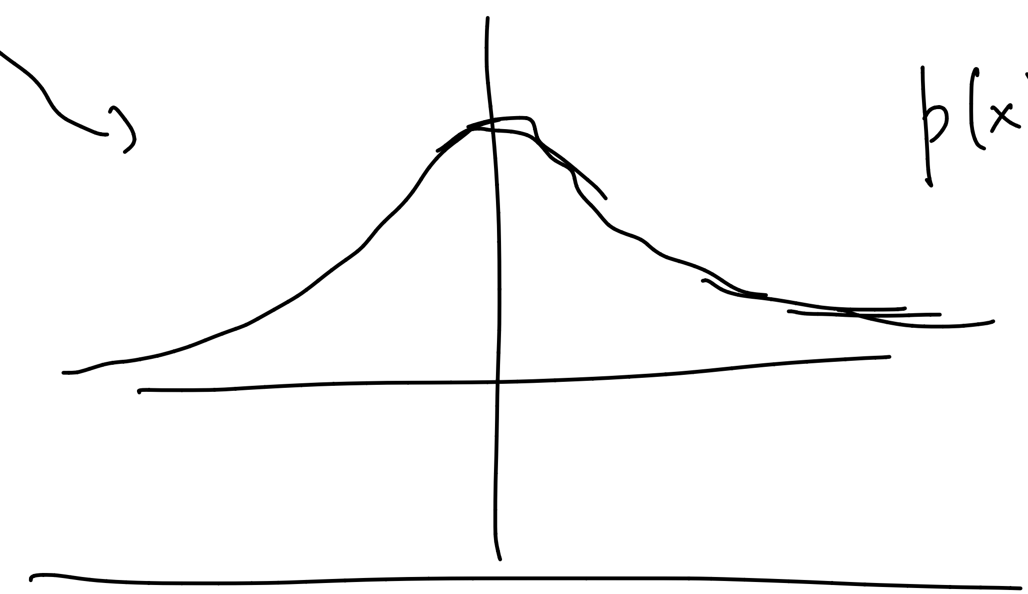


$$p(x) = \begin{cases} 1 & \text{if } x \in [0,1] \\ 0 & \text{o/wise} \end{cases}$$

- Unif [0,1]

$$\mathbb{E}[X] = \int_0^1 x \cdot dx = \frac{1}{2}.$$

- Gaussian N(0,1) $\to \mathbb{E}[X] = 0$



$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

- Discrete Bernoulli

$$\begin{cases} 1 & \text{w.p. } 1/3 \\ 0 & \text{w.p. } \frac{2}{3}. \end{cases}$$

$$\mathbb{E}[X] = \frac{2}{3} \cdot 0 + \frac{1}{3} \cdot 1 = \frac{1}{3}.$$

Expected running time

$A[0, \ldots, n-1]$.

- Recall problem from last class — $n/3$ of the A[*i*] are 0, find one *i*
- While not found: pick random index i and check if A[i]=0

Expected Running Time

(similar to tossing until seeing heads)
Running time is a random variable

X: running time of algorithm.

$$\mathbb{E}[X] = \sum_{x=1}^{\infty} x \cdot \Pr[X = x] = \sum_{x=1}^{\infty} x \cdot \frac{1}{3} \cdot \left(\frac{2}{3}\right)^{x-1}$$

$$\Pr[X = x] = \left(\frac{2}{3}\right)^{x-1} \cdot \frac{1}{3}.$$

$$= 3. \quad \left[\text{do it! verify..}\right].$$

Recurrence for the expectation

- What is the expected number of tosses of a fair coin before seeing heads?

Recurrence for the expectation

expected val

to see heads. $\overset{\alpha}{\parallel}$

- What is the expected number of tosses of a fair coin? $\longrightarrow$ α

F: the outcome of the first toss = heads?

- Useful identity for E[X]. For any event F,

$$\mathbb{E}[X] = \underbrace{p(F) \cdot \mathbb{E}[X|F] + (1 - p(F)) \cdot \mathbb{E}[X|\overline{F}]}_{\text{Conditional expectation.}}$$

Recurrence for the expectation

- What is the expected number of tosses of a fair coin?

F: event that first toss = heads.

$$\mathbb{E}[X \mid F] = 1.$$

$$\mathbb{E}[X \mid \bar{F}] \overset{?}{=} 1 + \alpha$$

expected # of tosses to see heads conditioned on first toss = tails.

$$\alpha = \frac{1}{2} \cdot 1 + \frac{1}{2}(1+\alpha)$$

$$\alpha = 1 + \frac{\alpha}{2}$$

$$\frac{\alpha}{2} = 1$$

$$\alpha = 2.$$

Quick sort

Problem: given unsorted array A[0, …, n-1], sort it.

- Pick a random element of A as the "pivot" → A[i]
- Partition A into sub-arrays *B* and *C*

B : all elements of A $\leq$ A[i]

C : all elts of A > A[i].

- Recursively sort *B*, *C*, and concatenate answers

sorted(B) A[i] sorted(C)

How long can it take?

$$T(n) = T(n-1) + n \rightarrow \text{prob} \approx \frac{2}{n}.$$

$$\rightsquigarrow T(n-2) + (n-1) + n$$

$$\rightsquigarrow \ddots$$

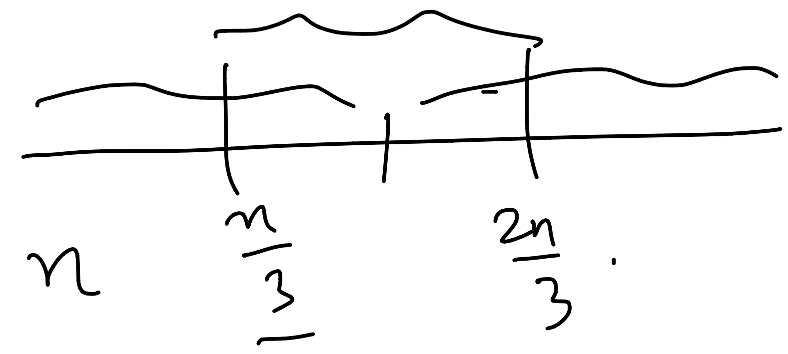
- Can it take n^2 time?

- If so, with what probability?

$\ddots$

What is the expected running time?

Good and bad pivots



$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + n$$

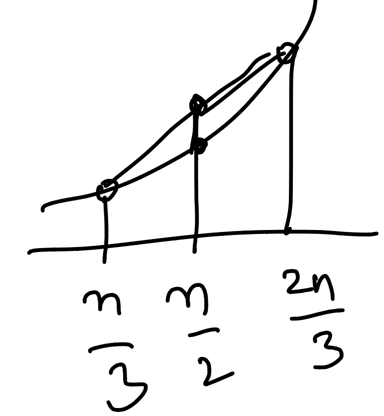
(exactly the same as merge sort) $\rightarrow$ good

$C \cdot n \log n$

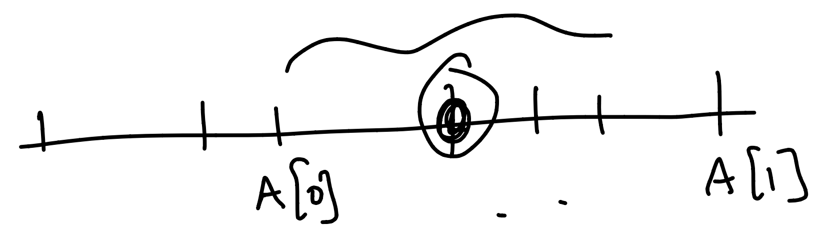
$$T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + n$$

- <u>Key observation:</u> it suffices to encounter log n "good pivots"

Expected running time



$f(n)$: expected running time of QuickSort on an array of length $= n$.



Running time $\equiv X$

F : event that pivot is between $\frac{n}{3}^{rd}$ & $\frac{2n}{3}^{rd}$ smallest elements.

Formula for conditional exp. :

$$f(n) = \frac{1}{3} \cdot \Big[\underbrace{f\left(\tfrac{2n}{3}\right) + f\left(\tfrac{n}{3}\right)}_{f\left(\frac{n}{2}\right) + f\left(\frac{n}{2}\right)} \Big] + \frac{2}{3} \cdot f(n-1) + n$$

$$f(n) = \frac{1}{3}\left[f\left(\frac{2n}{3}\right) + f\left(\frac{n}{3}\right)\right] + \frac{2}{3}\cdot f(n) + n$$

$$\Updownarrow$$

$$f(n) = f\left(\frac{2n}{3}\right) + f\left(\frac{n}{3}\right) + 3n$$

Akra-Bazzi / Guess-n-prove.

$$f(n) \leq c \cdot n \log n.$$

Recurrence for expected time

$$\mathbb{E}[X] = \underbrace{\Pr\left[\begin{matrix}\text{pivot is}\\ \text{smallest}\\ \text{elt}\end{matrix}\right]}_{\frac{1}{n}} \cdot \mathbb{E}\left[X \,\middle|\, \begin{matrix}\text{pivot is}\\ \text{smallest}\end{matrix}\right] + \underbrace{\Pr\left[\begin{matrix}\text{pivot is}\\ 2^{nd}\text{ smallest}\end{matrix}\right]}_{\frac{1}{n}} \cdot \mathbb{E}[X \mid \ldots]$$

$$+ \Pr\left[\begin{matrix}\text{pivot is } 3^{rd}\\ \text{smallest}\end{matrix}\right] \cdot \mathbb{E}\left[X \,\middle|\, \begin{matrix}\text{pivot is}\\ \text{third smallest}\end{matrix}\right] + \ldots$$

$$f(n) = \frac{1}{n}\Big[\big(f(n-1)\big) + \big(f(n-2)+f(1)\big) + \big(f(n-3)+f(2)\big) + \ldots \ldots\Big] + n.$$

$$= \frac{1}{n}\sum_{i=1}^{n-1}\left[f(i) + f(n-i-1)\right] + n$$

Solving the recurrence

Have shown: in expectation, quick-sort on array of length $= n$ takes time $\leq 4 n \log n$

From expectation to guarantee

Markov's inequality: let *X* be a non-negative random variable with expectation *C*. Then prob[X > tC] <= 1/t.

- Implication for quick sort?
- "Boosting probability"

Amplification by repetition