CS3000
6/13 - Tues.

Admin

- Long HW5 are 9pm
- Exam #2 Thurs → practice problem solns on piazza
- Optional rec. today
- no new HW today
- Short HW4 at next tues.
  - ↳ can't be resubmitted

Agenda

1. Expected run-time
2. Hiring problem
3. Randomized Quicksort

no more graphs :)
:)

1. Expected Run-Time

- Back to ~~an~~ analyzing algorithmic efficiency

  $\Theta(\sim)$ upper + lower bound

  $T(n) = \sim \sim \sim \sim$

  ↳ drop coeffs, lower order terms

We can have different bounds on best-case / worst-case

↳ us, usually

(ex) linear search

worst case $\Theta(n)$ ⇝ key not in A

best case $\Theta(1)$ ⇝ key first thing we look at

Best case: lucky
can't force luck

Worst-case: unlucky
- or, adversary
- but, we can make worst-case less likely (in some cases)

Develop an expected case, if worst-case is less likely

- scenario: input array is arranged for worst-case (adversary or bad luck)

- our goal: introduce randomization

}

1. Choose next element at random
2. Shuffle array before we start
   n! permutations
   all to be equally likely
3. Make decisions based on Random(0,1)
4. Reverse the array
   (if we know we have a worst-case array)
5. Check if already sorted

By intro'ing randomization
- can't change bound on worst-case run time
- But, we can have a new expected run-time
  (hopefully, better than worst-case run time)

Expected Value

↳ probability theory

Set of possible outcomes S (sample space)

each outcome s_i has some probability $Pr(s_i)$

Random Variables

↳ not random
  not a variable

X = value assoc. with outcome of an experiment (numeric)

Random variable can have expected value

$E[X]$ = expected value of X

average value of X if we run the experiment over and over and over

(today: avg run-time if we run the algo over+ over again)

$$\boxed{E[X] = \sum_{i=1}^{n} Pr(s_i) \cdot X_i}$$ → pr outcome × value of outcome

ex experiment: flip a coin 3 times

Rand. variable X = # tails in experiment

$S = \{TTT, TTH, THT, HTT, HTH, HHT, THH, HHH\}$

want: $E[X]$ = expected # tails in 3 flips

$= Pr(3 \text{ tails}) \cdot 3 + Pr(2 \text{ tails}) \cdot 2 + Pr(1 \text{ tail}) \cdot 1 + Pr(0 \text{ tails}) \cdot 0$

$= \frac{1}{8} \cdot 3 + \frac{3}{8} \cdot 2 + \frac{3}{8} \cdot 1 + \frac{1}{8} \cdot 0$

$= 1.5$

Indicator Random Variables

- look at each coin flip individually

- $X = 1$ if tails
  $= 0$ otherwise

same formula: $E[X] = 1 \cdot \frac{1}{2} + 0 \cdot \frac{1}{2}$
$= \frac{1}{2}$

apply idea to ith coin flip $E[X_i] = \frac{1}{2}$

$E[3 \text{ flips}] = E[X_1] + E[X_2] + E[X_3]$
$= \frac{1}{2} + \frac{1}{2} + \frac{1}{2}$
$= 1.5$ → same as annoying formula! :)

2. Hiring Problem

- Hiring some # of faculty
- We don't know in advance how many we'll hire
  - n = # candidates

```
HIRE(n)
  best = -∞
  for i = 1 to n
     • interview candidate i
     • if candidate i > best
          Best = candidate i
          hire candidate i
```

- interview is cheap
- hiring is expensive

We interview: n candidates

We hire...

Best-case (cheapest)

- one hire (candidate #1 best)

Worst-case (most expensive)

- n hires (candidates from worst to best)

↳ unlucky
or, adversary

Make worst-case scenario less likely...

option #2: shuffle candidates first

$\boxed{\text{Expected run-time}}$ with random candidate order?

X = # people hired

$X_i = 1$ if candidate i is hired
$\quad\;\; 0$ otherwise

} Ind. RV

What is probability of hiring cand. 1, 2, 3, ..., n?

<A, B, C, D> — random order

- Pr(hiring A) = 1
- Pr(hiring B) = 1/2
- Pr(hiring C) = 1/3
- Pr(hiring D) = 1/4
- ...

In general... $Pr(\text{hiring candidate } i) = 1/i$

$$E[X_i] = 1 \cdot 1/i + 0 \cdot {}^{i-1}/_i$$
$$= 1/i$$

In total... expected # of hires?

$$E[X] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} 1/i = \ln(n) + \Theta(1)$$

$$\approx \Theta(\lg n)$$

10:57

3. Randomized Quicksort

Remember that...

```
QSORT (A)
  if p<r
    q = PARTITION(A, p, r)
    QSORT (A, p, q-1)
    QSORT (A, q+1, r)
```

→ choose $A[r]$ as pivot
- smaller on left
- bigger on right

worst case:

- unlucky pivot, or adversary
- everything on one side or the other

$$T(n) = \underbrace{T(n-1)}_{\text{everything on one side}} + \underbrace{\Theta(n)}_{\text{cost of partition}}$$

$= \Theta(n^2)$

$\boxed{\text{Best case:}}$

- lucky pivot
- evenly split array

$$T(n) = T(n/2) + T(n/2) + \Theta(n)$$
$$= \Theta(n \lg n)$$

We also get best case with any constant split

$$T(n) = T\left(\frac{9n}{10}\right) + T\left(\frac{n}{10}\right) + \Theta(n)$$
$$T(n) = T\left(\frac{2n}{3}\right) + T(n/3) + \Theta(n)$$

↳ $\Theta(n \lg n)$

$\boxed{\text{Apply randomization \#2} \quad \bullet \text{ choose random pivot}}$

→ Instead of A[r]

$\boxed{\text{choose random } i \text{ } [p..r] \quad A[i] \text{ is pivot}}$

In general... $\boxed{T(n) = T(\text{split } 1) + T(\text{split } 2) + \Theta(n)}$

- subtract = worst case
- divide = best case

$\boxed{\text{Balanced partition}}$ n/4 to 3n/4 : $\boxed{\text{final position of pivot}}$

[~ Pivot ~ ~ ~]
[— — pivot ~ ~] → balanced
[~ ~ ~ pivot ~]

[pivot ~ ~ ~ ~ ~] → unbalanced
[~ ~ ~ ~ pivot]

- all positions of pivot are equally likely
  (1..n) possible locations the pivot could end up

- $n/4$ to $3n/4$ is half the positions

$\nearrow$ partition

balanced: $T(n) \le T(3n/4) + T(n/4) + cn$

unbalanced: $T(n) \le T(n-1) + cn$
$< T(n) + cn$

- each scenario has 50% prob

total ...

$$T(n) \le \underbrace{\tfrac{1}{2}\cdot(T(n) + cn)}_{\text{unbal.}} + \underbrace{\tfrac{1}{2}\cdot(T(3n/4) + T(n/4) + cn)}_{\text{balanced}}$$

$$T(n) \le \tfrac{1}{2}T(n) + \tfrac{1}{2}cn + \tfrac{1}{2}T(3n/4) + \tfrac{1}{2}T(n/4) + \tfrac{1}{2}cn$$

$$\tfrac{1}{2}T(n) \le \underline{\tfrac{1}{2}cn + \tfrac{1}{2}cn} + \tfrac{1}{2}T(3n/4) + \tfrac{1}{2}T(n/4)$$

$$\tfrac{1}{2}T(n) \le cn + \tfrac{1}{2}T(3n/4) + \tfrac{1}{2}T(n/4)$$

$$T(n) \le 2cn + T(3n/4) + T(n/4)$$

$O(n \lg n)$!! :)

so much better than n^2 !!!