COP 3503 2/15/22

CS1 Stuff - Posted

(1) Real Def of O, Ω, Θ

Two equiv defs

(a) for all c_0 blah, blah... NON-INTUITIVE

(b) limit definition

$O(f(n))$ is defining a set of functions.

$O(n)$, set of all function $f(n)$ s.t.

$$\lim_{n \to \infty} \frac{f(n)}{n} = c, \text{ for constant } c.$$

Set of all functions $g(n)$ that are $\in O(f(n))$ is the set of all functions s.t.

$$\lim_{n \to \infty} \frac{g(n)}{f(n)} = c, \text{ for constant } c.$$

Examples

$n \in O(n)$

$10n \in O(n)$

$10^6 n - \lg n \in O(n)$

↓

If $g(n) \in O(f(n)) \Rightarrow$

Intuitively $O(n)$ is set of functions that is no bigger than cn for some const c, as n grows large.

$g(n) \leq f(n)$ [SORT OF]

People will write $10n = O(n)$. It just means what I have above.

So when people provide a big-oh run-time, it's an upper bound. (Doesn't have to be a tight upper bound.)

Insertion Sort is an $O(n^5)$ algorithm $\Rightarrow$ TRUE
↳ not a tight upper bound.

$n^6 \notin O(n^5)$ but $n^4 \in O(n^5)$

if $f(n) \in O(g(n))$, then $cf(n) \in O(g(n))$.
↳ any constant.

$\lg n^2 = 2\lg n \in O(\lg n)$

$g(n) \in \Omega(f(n))$ iff $\lim_{n\to\infty} \frac{g(n)}{f(n)} > 0$ — $f(n)$ is a lower bound on $g(n)$

$g(n) \in \Omega(f(n)) \Rightarrow g(n) \geq f(n)$ sort of

$g(n) \in \Theta(f(n))$ iff $\lim_{n\to\infty} \frac{g(n)}{f(n)} = c$, const and $c > 0$.

Insertion Sort is $\Omega(n)$ and $O(n^2)$

If an alg runs in $\Omega(f(n))$ and $O(f(n))$ it's also $\Theta(f(n))$. IE. Merge Sort on n items is $\Theta(n\lg n)$.

Chart of Growth log stuff slow exception $\ln e^n = n$

polys

exponential

$\log_2 n \in \Theta(\log_4 n)$ but $2^n \notin \Theta(4^n)$...

be careful w/logs, exponents.

Recurrence Relations

Master Theorem

$$T(n) = AT\left(\frac{n}{B}\right) + O(n^k)$$

Case $B^k < A \Rightarrow O(n^{\log_B A})$

Case $B^k = A \Rightarrow O(n^k \lg n)$

Case $B^k > A \Rightarrow O(n^k)$

Expectation Definition

Discrete Random Variable

$$X = \sum \begin{matrix} 1 & 1/10 \\ 2 & 1/10 \\ 3 & 1/10 \\ 4, \text{ w prob } 1/10 \\ 5, \text{ w prob } 1/10 \\ 6, \text{ w prob } \frac{1}{2} \end{matrix}$$

avg roll per die.

$$E(X) = \sum_{x \in X} x \cdot p(x)$$

$$\frac{1}{10}\times 1 + \frac{1}{10}\times 2 + \frac{1}{10}\times 3 + \frac{1}{10}\times 4 + \frac{1}{10}\times 5 + \frac{1}{2}\times 6 = 4{,}5$$

Binomial Thm / Probability Distribution

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^k y^{n-k}$$

If I repeat some trial n times and each time my probability of success is p, then the probability of exactly k successes out of n is

$$\binom{n}{k} p^k (1-p)^{n-k}$$

$(1-p)^{n-k}$ → Same for failures.

$\binom{n}{k}$ ↓ # orderings of k successes $n-k$ failures

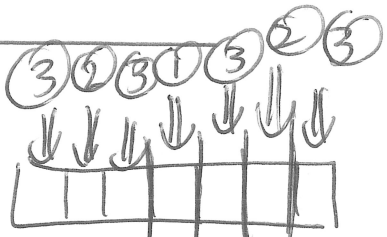
p^k → probability of the k successes in their specific slots

Note: Expected Runtime → deal w/ sums a lot!

Problem #1: Avg Case Binary Search

Simplifications:

(1) $n = 2^k - 1$



(2) We search for an item always in the array and each is equally likely.

# Comparisons	# elements found	probability
1	1	$\frac{1}{n}$
2	2	$\frac{2}{n}$
3	4	$\frac{4}{n}$
4	8	$\frac{8}{n}$
m	2^{m-1}	$\frac{2^{m-1}}{n}$
k-1	2^{k-2}	$\frac{2^{k-1}}{n}$
k	2^{k-1}	

$$\text{Avg Run Time} = \frac{1}{n}\times 1 + \frac{2}{n}\times 2 + \frac{4}{n}\times 3 + \frac{8}{n}\times 4 - + \frac{2^{k-1}}{n}\times k$$

$$\frac{1}{n}S = \sum_{i=0}^{k-1} \frac{2^i}{n}\times(i+1) = \sum_{i=1}^{k} \frac{2^{i-1}}{n}\times i = \frac{1}{n}\sum_{i=1}^{k} i\cdot 2^{i-1}$$

$$S = 1\times 2^0 + 2\times 2^1 + 3\times 2^2 + 4\times 2^3 + \cdots \quad k\cdot 2^{k-1}$$

$$-2S = \quad 1\times 2^1 + 2\times 2^2 + 3\times 2^3 + \quad (k-1)\cdot 2^{k-1} + k\cdot 2^k$$

$$-S = 1\times 2^0 + 1\times 2^1 + 1\times 2^2 \qquad + 1\cdot 2^{k-1} - k\cdot 2^k$$

$$S = k\cdot 2^k - \left(2^0 + 2^1 + 2^2 + \cdots 2^{k-1}\right)$$

$$S = k\cdot 2^k - \frac{2^k - 1}{2-1}$$

$$= k\cdot 2^k - (2^k - 1)$$

$$= k\cdot 2^k - 2^k + 1$$

$$= 2^k(k-1) + 1 \qquad n = 2^k - 1 \qquad \log_2(n+1) = k$$

$$S = (n+1)(\log_2(n+1) - 1) + 1$$

$$\text{Avg Case Run Time} = \frac{S}{n} = \frac{(n+1)(\log_2(n+1) - 1) + 1}{n}$$

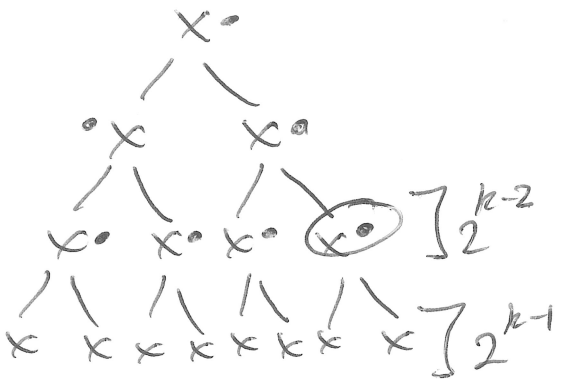
$$= O(\lg n)$$

Make Heap with n items (worst case)

Assume the following

(1) $n = 2^k - 1$ (full heap of k levels)

(2) Worst case behavior, we swap all the way down to the bottom row, every time.



# Items	max swaps
2^{k-2}	1
2^{k-3}	2
2^{k-4}	3
2^{k-5}	4
⋮	
1	k-1

$$S = \underline{2^{k-2} \times 1} + \underline{2^{k-3} \times 2} + 2^{k-4} \times 3 + \cdots \quad 2^1 \times (k-2) + 2^0 (k-1)$$

$$-\frac{S}{2} = \qquad 2^{k-3} \times 1 + 2^{k-4} \times 2 \qquad 2^1 \times (k-3) + 2^0 (k-2) + \tfrac{1}{2}(k-1)$$

$$\frac{S}{2} = 2^{k-2} + 2^{k-3} + 2^{k-4} \cdots \quad + 2^1 + 2^0 - \tfrac{1}{2}(k-1)$$

$$S = 2^{k-1} + 2^{k-2} + \cdots 2^1 - (k-1)$$

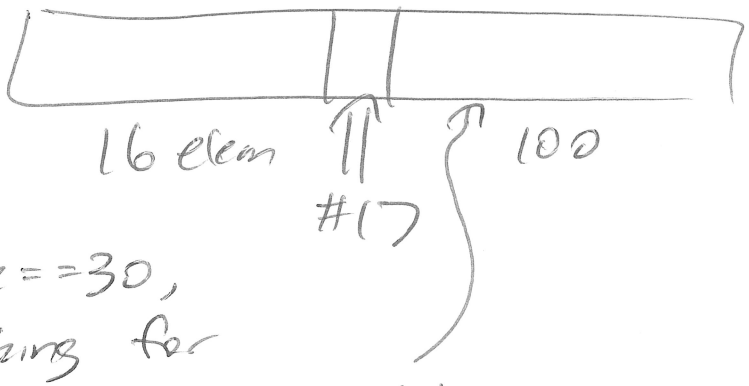
$$= \underbrace{2^{k-1} + 2^{k-2} + \cdots 2^1 + 1}_{n} - k$$

$$= n - k = O(n).$$

Quick Select

Problem: find the k^{th} smallest item in an array.

QuickSelect Alg: Run partition (same as function from QuickSort)



e.g. if $k==30$,
I'd be looking for
13^{th} smallest item on right

if $k \leq 16$
go left
if $k == 17$
found
if $k > 17$
go right.

Assume each input of k is equally likely

Assume each split from partition is equally likely.

Partition Item	prob	
idx 0	$\frac{1}{n}$	$\frac{1}{n}$ done $\frac{n-1}{n}$ look array n-1
idx 1	$\frac{1}{n}$	$\frac{1}{n}$ done, $\frac{1}{n}$ look array 1, $\frac{n-2}{n}$ array n-2
idx 2	$\frac{1}{n}$	$\frac{2}{n}$ look array 2, $\frac{n-3}{n}$ array n-3
idx n-1	$\frac{1}{n}$	

Quick Select Avg Case Run Time

n steps for partition

$$T(n) = \frac{2}{n}\left[\frac{1}{n}T(1) + \frac{2}{n}T(2) + \cdots + \frac{n-1}{n}T(n-1)\right] + h$$

$$n^2 T(n) = 2\left(T(1) + 2T(2) + 3T(3) \cdots + (n-1)T(n-1)\right) + n^3$$

$$-\,(n-1)^2 T(n-1) = 2\left(T(1) + 2T(2) + 3T(3) \cdots + (n-2)T(n-2)\right) + (n-1)^3$$

$$n^2 T(n) - (n-1)^2 T(n-1) = (n-1)T(n-1) + (2n-1)(3n^2-3n+1)$$

$$n^2 T(n) = n(n-1)T(n-1) + (2n-1)(3n^2-3n+1), \text{ sorry, skipped algebra}$$

$$nT(n) = (n-1)T(n-1) + \left(2-\frac{1}{n}\right)\left(3n-3+\frac{1}{n}\right)$$

Let $S(n) = nT(n)$

$$S(n) = S(n-1) + \left(2-\frac{1}{n}\right)\left(3n-3+\frac{1}{n}\right)$$

$$S(n) = 2n \sum_{i=1}^{n} \frac{1}{i} = 2n \cdot H_n, \; H_n \sim \ln n$$

$$\leq \sum_{i=1}^{n} 3i = \frac{3n(n+1)}{2}$$

Drop $-(3-\frac{1}{n})$

$$T(n) = \frac{S(n)}{n} \leq \frac{\frac{3n(n+1)}{2}}{n} = \frac{3}{2}(n+1) \in O(n)$$