1/18/2018 CS2

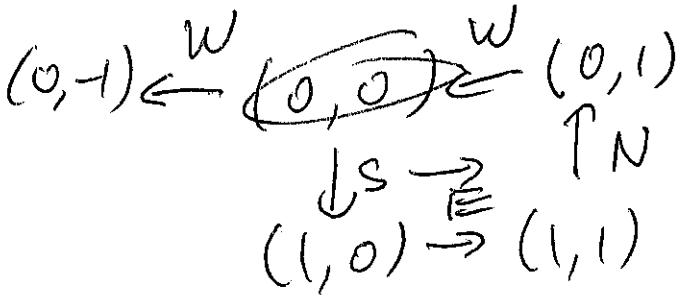
① Go over asgn protocol
stdin, stdout

② Alg Analysis
Formal Def O, Ω, Θ

③ Theoretical Analysis Example
- Binary Search

④ Empirical Run Time Analysis
- MCSS problem

1/18/18 (2)



dir map

N → 0

E → 1

S → 2

W → 3

"0,0" → 0

"0,0" → 1

"1,1" → 2

"0,1" → 3

Process "0,0"

"0,-1" → 1

cnt = 0;

Big-Oh Formal Def

$f(n) \in O(g(n))$ iff

There exist a constants n_0, and C such that

for all $n > n_0$, $f(n) \leq C \cdot g(n)$, for some constant C.

$4n^2 + 3n \in O(n^2)$ True

$2n^2 \in O(2^n)$ True

→ let $n_0 = 1$, $c = 8$

for all $n > 1$, and $c = 8$,

$f(n) = 4n^2 + 3n \leq 8n^2$

$4n^2 + 3n \leq 7n^2$ (when $n > 1$)

$\leq 8n^2$

→ $n_0 = 10$ $c = 1$

for all $n > 10$, $2n^2 \leq 2^n$

at 10 $2n^2 = 200$

$2^n = 1024$

This grows faster

$f(n) \leq g(n)$

$f(n)$ doesn't grow faster than $g(n)$ by more than a constant amount.

Calc Defn

1/18/18 (4)

$f(n) \in O(g(n))$ iff

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

$$\lim_{n\to\infty} \frac{2n^2}{2^n} = 0 \quad \checkmark$$

$$\lim_{n\to\infty} \frac{4n^2+3n}{n^2} = 4 \quad \checkmark$$

$f(n) \in \Omega(g(n))$ iff

There exist consts n_0, c

for all $n > n_0$ $f(n) \geq c \cdot g(n)$

$4n^2+3n \in \Omega(n^2)$

$4n^2+3n \in \Omega(\lg n)$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} > 0$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c, \text{ const } c \neq 0$$

Θ : $f(n) \in \Theta(g(n))$ iff

$f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$

Insertion Sort runs in $O(n^2)$ when sorting n numbers.

every time you run it it'll take fewer than cn^2 steps for some c.

best case n ~~steps~~ comparisons

worst case $\frac{1}{2}n^2$ comparisons

Technically inaccurate to say runs in $O(n^2)$ because $n \notin O(n^2)$.

Merge Sort runs in $\Theta(n \lg n)$ time.

Quick Sort is $O(n^2), \Omega(n \lg n)$

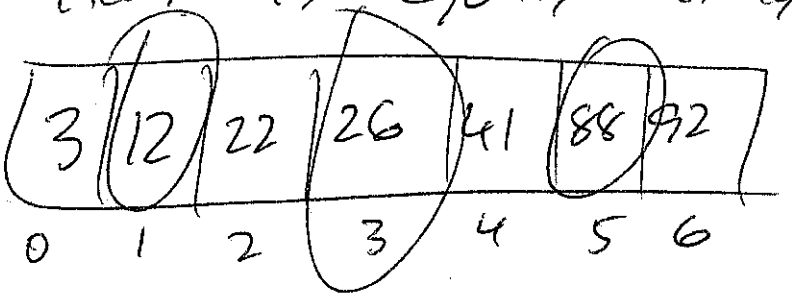
Avg Case QS is $\Theta(n \lg n)$

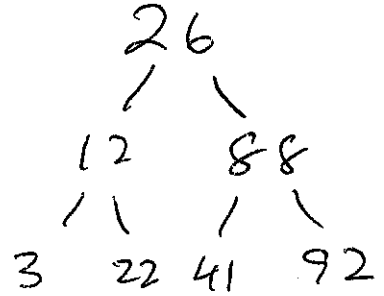
Avg Case Analysis Bin Search

(1) Assume array size is $n = 2^k - 1$.

(2) Assume we're always searching for an item in the array.

(3) each item is equally likely to search.





26 1 try
12 2 tries
88 2 tries

$n = 7$ 3, 22, 41, 92 3 tries

$$E(\text{time}) = \frac{1}{7} \times 1 + \frac{2}{7} \times 2 + \frac{4}{7} \times 3 = \frac{1}{7} + \frac{4}{7} + \frac{12}{7} = \boxed{\frac{17}{7}}$$

General Case $n = 2^k - 1$:

$$E(\text{time}) = \frac{1}{2^k-1} \times 1 + \frac{2}{2^k-1} \times 2 + \frac{4}{2^k-1} \times 3 + \frac{8}{2^k-1} \times 4 + \frac{16}{2^k-1} \times 5 + \ldots + \frac{2^{k-1}}{2^k-1} \times k$$

let S_{2^k}

$$= \left(\frac{1}{2^k-1}\right)\left(1\times1 + 2\times2 + 4\times3 + 8\times4 + 16\times5 + \ldots 2^{k-1}\cdot k\right)$$

1/18/18

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

$$S = 1\times 1 + 2\times 2 + 3\times 4 + 4\times 8 + 5\times 16 + \cdots + k\cdot 2^{k-1}$$

$$-2S = \quad 1\times 2 + 2\times 4 + 3\times 8 + 4\times 16 \quad (k-1)\cdot 2^{k-1} + k\cdot 2^{k}$$

$$-S = 1\times 1 + 1\times 2 + 1\times 4 + 1\times 8 + 1\times 16 + 1\cdot 2^{k-1} - k\cdot 2^{k}$$

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

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

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

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

$$n = 2^{k-1}$$

$$k = \log_2(n+1)$$

$$E(\text{time}) = \frac{(k-1)\cdot 2^{k}+1}{2^{k}-1} \sim (k-1)$$

$$E(\text{time}) = \frac{1}{2^{k}-1}\times 1 + \frac{2^{k}-2}{2^{k}-1}\,E\left(2^{k-1}-1\right)$$

$$E(n) = \frac{1}{n} + \frac{n-1}{n}\,E(n/2)$$

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

$$= \frac{n-1}{n}\left(\frac{\frac{n}{2}-1}{\frac{n}{2}} E\left(\frac{n}{4}\right) + \frac{2}{n}\right)$$

= looks even uglier...
will stop now.