6.006 Fall 2011
L 1

http://courses.csail.mit.edu/6.006

Administrivia

Course overview

"Peak finding" problem
- 1D version
- 2D version

Course Overview

- Efficient procedures for solving problems on large inputs (e.g., US highway map, human genome)
- Scalability
- Classic data structures and elementary algorithms (CLRS text)
- Real implementations in Python
- Fun problem sets

Content

8 modules each with motivating problem and problem set(s) (except last)

Algorithmic thinking : Peak finding
Sorting & Trees : Event simulation
Hashing : Genome comparison
Numerics : RSA encryption
Graphs : Rubik's cube
Shortest Paths : Caltech → MIT
Dynamic Programming : Image compression
Advanced Topics

TENTATIVE

PEAK FINDER

One-dimensional version

1	2	3	4	5	6	7	8	9
a	b	c	d	e	f	g	h	i

a-i are numbers

Position 2 is a peak if and only if
$b \geq a$ and $b \geq c$

Position 9 is a peak if $i \geq h$

Problem: Find a peak if it exists.*

6	7	4	3	2	1	4	5

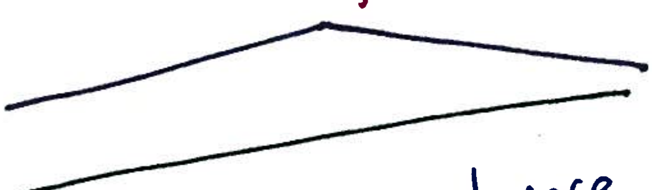
* Does it always exist?

STRAIGHTFORWARD ALGORITHM

Start from left

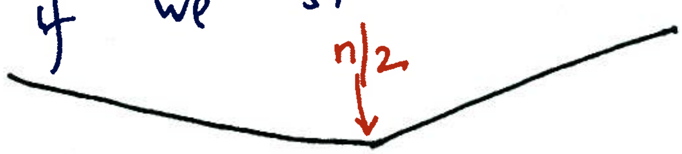
might be peak

Look at n/2 elements
Could look at n elements



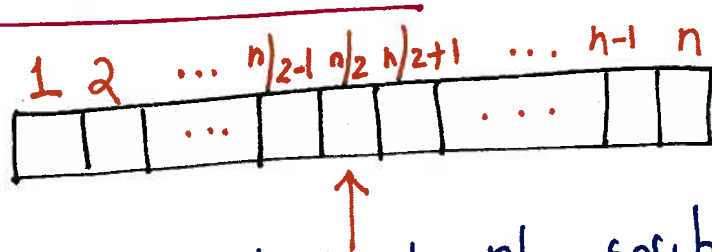
$\Theta(n)$ complexity worst case

What if we start in the middle?



Look at n/2 elements

Can we do better?



Divide & Conquer

Look at $n/2$ position

If $a[n/2] < a[n/2-1]$ then only look at left half $1 \ldots n/2-1$ to look for peak

Else if $a[n/2] < a[n/2+1]$ then only look at right half $n/2+1 \ldots n$ to look for peak

Else $n/2$ position is a peak

WHY? $a[n/2] \geq a[n/2-1]$
$a[n/2] \geq a[n/2+1]$

What is the complexity?

$$T(n) = T(n/2) + \Theta(1)$$
$$= \Theta(1) + \cdots + \Theta(1) \text{ } (\log_2 n \text{ times})^*$$
$$T(n) = \Theta(\log_2 n)$$

To compare $a[n/2]$ to neighbors

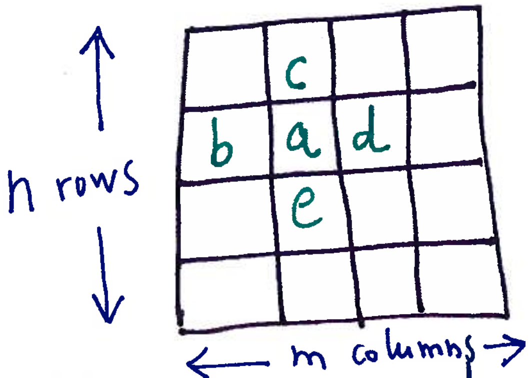
$n = 1{,}000{,}000$ $\quad \Theta(n)$ algo $\quad$ 13 s in python impl

$\Theta(\log n)$ algo $\quad$ 0.001 s

Argue that the algorithm is correct

* In order to sum up the $\Theta(1)$'s as we do here, we need to find a constant that works for all.

2-Dimensional Version

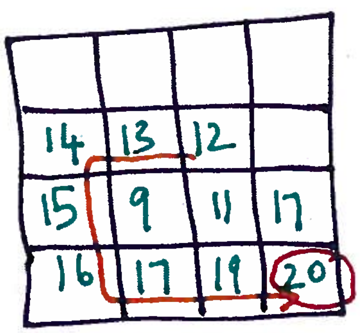


a is 2D peak iff

$a \geq b$, $a \geq d$, $a \geq c$, $a \geq e$

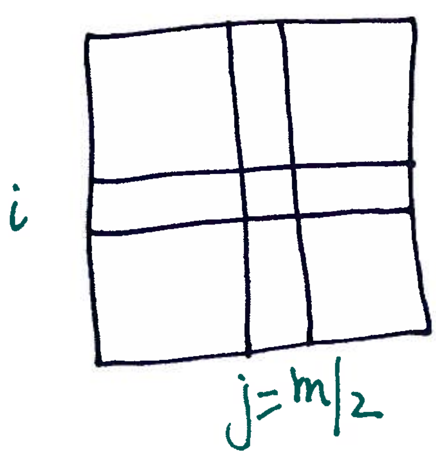
Greedy ascent algorithm : $\Theta(nm)$ complexity

$\Theta(n^2)$ algorithm if $m = n$



○ peak

Extend 1D divide & conquer to 2D : Attempt #1



Pick middle column $j = m/2$

Find a 1D peak at i, j

Use (i, j) as a start point on row i to find 1D-peak on row i

Attempt #1 Fails

Problem: 2D peak may not exist on row i

		10	
14	13	12	
15	9	11	
16	17	19	20

end up with 14 which is not a 2D peak

Attempt # 2

Pick middle column $j = m/2$

Find global maximum on column j at (i, j)

Compare $(i, j-1)$, (i, j), $(i, j+1)$

Pick left cols if $(i, j-1) > (i, j)$

(similarly for right)

(i, j) is a 2D-peak if neither condition holds

Solve the new problem with half the number of columns

When you have a single column, find global maximum and you're done

EXAMPLE OF ATTEMPT #2

10	8	10	10
14	13	12	11
15	9	11	21
16	17	19	20

pick this column

17 global maximum for column

go with

10	10
12	11
11	21
19	20

pick this column

19 is global maximum for column

10
11
21
20

find 21

COMPLEXITY OF ATTEMPT #2

n rows, m columns

$$T(n, m) = T(n, m/2) + \Theta(n)$$

to find global maximum on a column (n rows)

$$T(n,m) = \underbrace{\Theta(n) + \ldots \Theta(n)}_{\log m} = \Theta(n \log m)$$

$$= \Theta(n \log n) \text{ if } m = n$$

Q: What if we replaced global maximum with 1D-peak in Attempt #2? Would that work?