RECITATION 2 09/16

PLAN:

1. CORRECTNESS & COMPLEXITY OF $O(n \log n)$ ALGORITHM
2. $O(n)$
3. ANOTHER $O(n)$ ALGORITHM
4. COUNTEREXAMPLE I
5. COUNTEREXAMPLE II

ALGORITHM 1: $O(n \log n)$

(slightly different from the one in lecture)

$$A = \begin{array}{|c|c|c|c|c|} \hline 3 & 5 & 3 & 5 & 7 \\ \hline 7 & 4 & 7 & 2 & 8 \\ \hline 15 & 3 & 3 & 3 & 1 \\ \hline 11 & 1 & 7 & 12 & 2 \\ \hline 8 & 11 & 1 & 5 & 2 \\ \hline \end{array}$$

$$\text{max} \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$A' = \begin{array}{|c|c|c|c|c|} \hline 15 & 11 & 7 & 12 & 8 \\ \hline \end{array}$$

1. CONSIDER PROJECTION A' WHERE EACH ENTRY IS THE MAXIMUM OF THE CORRESPONDING COLUMN OF A
2. FIND ~~MA~~ 1D-PEAK IN A' USING $O(\log n)$ ALG
3. FIND ITS OCCURRENCE IN THE CORRESPONDING COLUMN OF A

IT MUST BE A 2D-PEAK. WHY?

2

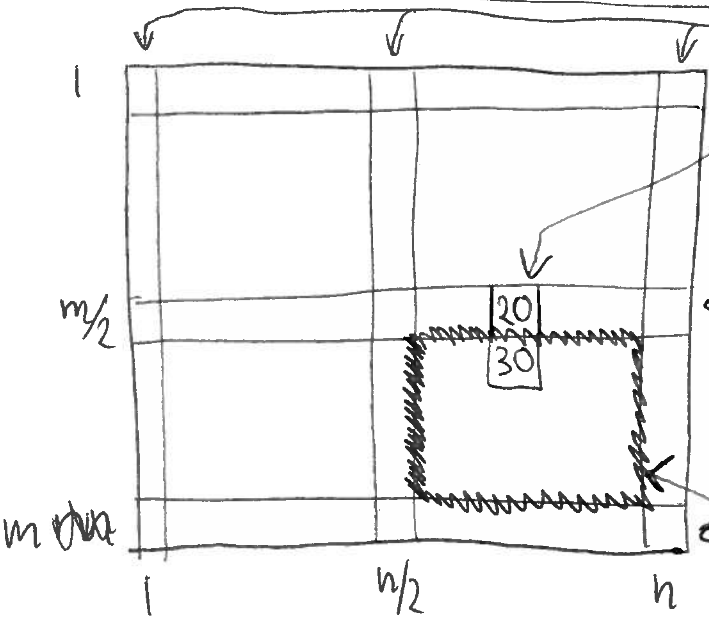
COMPUTING A' EXPLICITLY : $\Theta(n^2)$ TIME ← TOO MUCH

COMPUTING A SINGLE ENTRY OF A' : $\Theta(n)$ TIME

1D-PEAK LOCATOR ONLY LOOKS AT $O(\log n)$ ENTRIES OF A'

IT SUFFICES TO COMPUTE ONLY THEM! $\underbrace{O(n \log n)}_{\text{TOTAL COMPUTATION TIME}}$

ALGORITHM 2: $O(n)$ FROM THE LECTURE



1. COMPUTE MAXIMUM OF 3 ROWS AND 3 COLUMNS — $O(n+m)$ TIME
2. IF IT IS A 2D-PEAK, DONE
3. OTHERWISE, RECURSE ON THE QUADRANT WITH A HIGHER NEIGHBOR

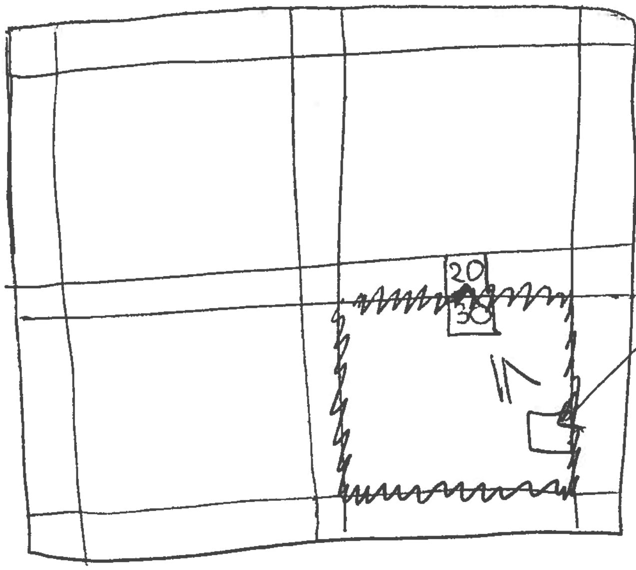
WANT TO PROVE BY INDUCTION A STRONGER STATEMENT:

THE ALGORITHM FINDS A 2D-PEAK THAT IS $\geq$ ANYTHING ON THE BOUNDARY

3

→ CLEARLY WORKS FOR SMALL MATRICES

→ FOR LARGE ONES:



2D-PEAK IN

FOUND BY RECURSIVE CALL

THE PEAK MUST BE GREATER THAN ANYTHING AROUND SO IS A 2D-PEAK IN THE ENTIRE MATRIX

TIME COMPLEXITY $T(n,m)$:

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

$$= O(n+m) + O\left(\frac{n+m}{2}\right) + O\left(\frac{n+m}{4}\right) + \cdots + O(1)$$

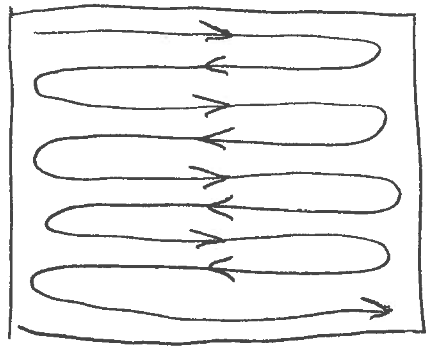
GEOMETRIC SEQUENCE

$$= O(n+m)$$

④

ALGORITHM 3: $O(n)$ IN EXPECTATION

RECALL: GREEDILY ASCENDING CAN TAKE $\Omega(n^2)$ TIME



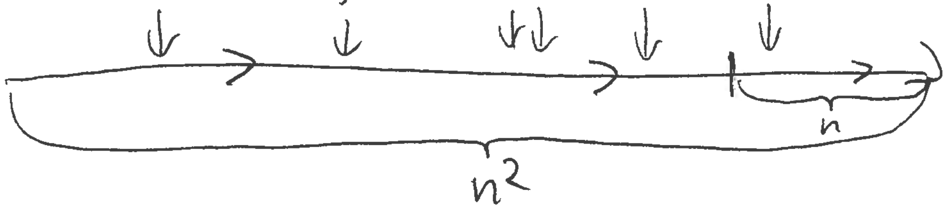
IS THERE A WAY TO FIX IT?

IDEA 1: START FROM A RANDOM LOCATION

→ CAN STILL TAKE $\Omega(n^2)$ TIME MOST OF THE TIME

IDEA 2:
- PICK n RANDOM LOCATIONS
- FIND THEIR MAXIMUM
- GREEDILY ASCEND FROM THERE

INTUITION: IF THERE IS A VERY LONG $\left(\Omega(n^2)\right)$ INCREASING PATH, VERY LIKELY TO DRAW A LOCATION WITHIN THE LAST $O(n)$ LOCATIONS



⑤ ANALYSIS:

→ CONSIDER THE SORTED SEQUENCE OF ALL THE NUMBERS:

$$a_1 \leq a_2 \leq \ldots \leq a_{n^2-2} \leq a_{n^2-1} \leq a_{n^2}$$

→ IF THE LOCATION CONTAINING a_i IS SELECTED THEN CAN ASCEND FOR AT MOST $n^2 - i$ STEPS

→ PROBABILITY ASCENDS FOR $\geq s$ STEPS

$\leq$ NO NUMBER IN $a_{n^2-s+1} \ldots a_{n^2}$ SELECTED

$\leq$ (~~RAND~~ PROBABILITY RANDOM INTEGER IN $\{1, \ldots, n^2\}$ DOESN'T BELONG TO $\{n^2-s+1, \ldots, n^2\}$)^n

$$\leq \left(1 - \frac{s}{n^2}\right)^n \leq e^{-\frac{s}{n^2} \cdot n} = e^{-s/n}$$

→ EXPECTED NUMBER OF STEPS

$$\leq \sum_{i=1}^{n} in \cdot \Pr[\text{walks for } (i-1)n+1 \ldots in \text{ steps}]$$

$$\leq n + \sum_{i=2}^{n} in \cdot \Pr[\text{walks for} \geq (i-1)n \text{ steps}]$$

$$\leq n + n \cdot \underbrace{\sum_{i=2}^{n} i \cdot e^{-(i-1)}}_{O(1)} = O(n)$$

HOW ABOUT 3 DIMENSIONS?

CAN GENERALIZE THE ALGORITHMS AS FOLLOWS

ALGORITHM 1: $O(n^2 \log n)$ ← PROJECT INTO 1D

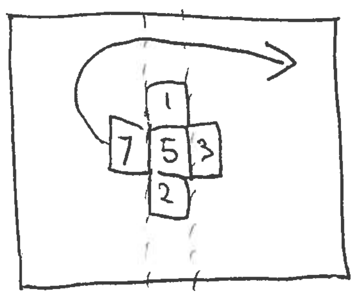
ALGORITHM 2: $O(n^2)$ ← PROJECT INTO 2D

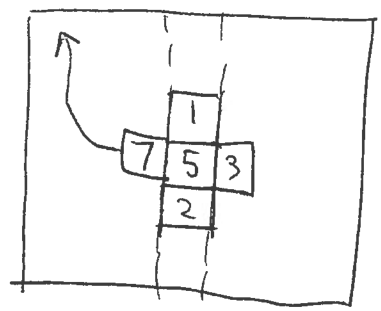
ALGORITHM 3: $O(n^{1.5})$ ← START FROM THE BEST OUT OF $n^{1.5}$ RANDOM LOCATIONS

COUNTEREXAMPLE I

CAN USE ANY PEAK INSTEAD OF THE HIGHEST PEAK FOR COLUMNS IN ALGORITHM 1?

NO:

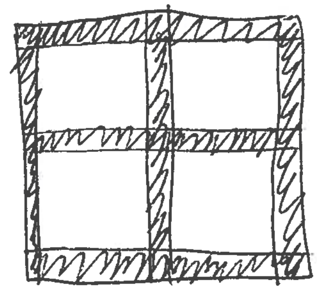




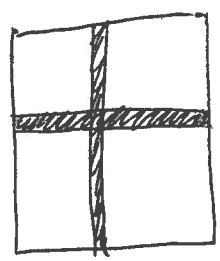
THERE MAY BE ONLY ONE PEAK AND WE DON'T LEARN IN WHICH HALF IT LIES, SO WE CAN'T RECURSE

COUNTEREXAMPLE II

QUESTION: ALGORITHM 2 LOOKS AT 3 COLUMNS & 3 ROWS:

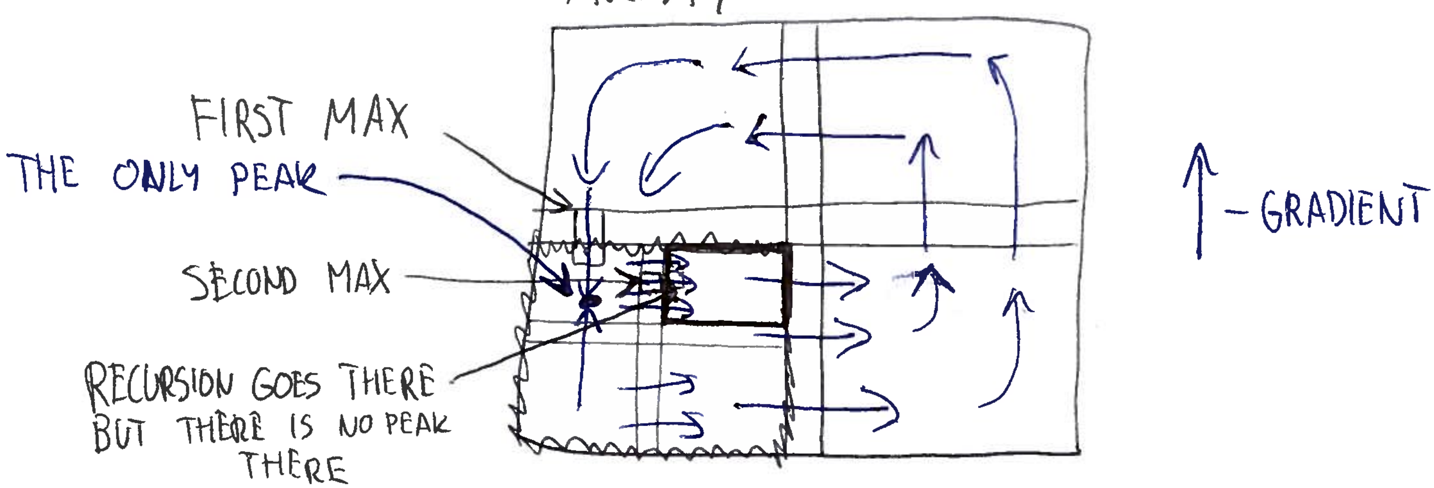


IS IT POSSIBLE TO LOOK AT JUST ONE ROW & ONE COLUMN



AND RECURSE BASED ON THEM?

ANSWER: NO. CAN RECURSE ON A SQUARE THAT HAS NO 2D-PEAK IN THE ENTIRE ARRAY, EVEN THOUGH HAS A 2D-PEAK FOR A SUB ARRAY



PSET 1 SHOWS HOW TO FIX THIS SOLUTION