Recitation (III): Divide-and-Conquer

▷ Solving Recurrence

- $T(n) = 4T(n/3) + O(n)$ $\Rightarrow$ $T(n) = O(n^{\log_3 4})$
- $T(n) = 3T(n/3) + O(n)$ $\Rightarrow$ $T(n) = O(n \log n)$
- $T(n) = 4T(n/2) + O(n^2\sqrt{n})$ $\Rightarrow$ $T(n) = O(n^2\sqrt{n})$
- $T(n) = 8T(n/2) + O(n^3)$ $\Rightarrow$ $T(n) = O(n^3 \log n)$

Exmp (I): Integer Multiplication

Given two n-digit integers, output their product. Design a $n^{\log_2 3}$-time algorithm to solve the problem. Notice that you can not multiple two big integers directly using a single operation.

Exmp:

$$x = 156978 \qquad\qquad X = [8, 7, 9, 6, 5, 1]$$
$$\Longrightarrow$$
$$y = 723541 \qquad\qquad Y = [1, 4, 5, 3, 2, 7]$$

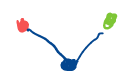
Equiv to the polyn multiplication problem:

$$p(a) = 8 + 7a + 9a^2 + 6a^3 + 5a^4 + a^5 \quad \Rightarrow \quad x = p(10)$$

$$q(a) = 1 + 4a + 5a^2 + 3a^3 + 2a^4 + 7a^5 \quad \Rightarrow \quad y = q(10)$$

Exmp (II). Local Minimum in 1D

Given an array $A[1 \,..\, n]$ of n **distinct** numbers, we say that some index $i \in \{1, 2, 3 \cdots, n\}$ is a local minimum of A, if $A[i] < A[i-1]$ and $A[i] < A[i+1]$ (we assume that $A[0] = A[n+1] = \infty$). Suppose the array A is already stored in memory. Give an $O(\lg n)$-time algorithm to find a local minimum of A.

$A[mid-1] > A[mid] < A[mid+1]$  . Done.

$A[mid-1] > A[mid] > A[mid+1]$ recurse into right half.

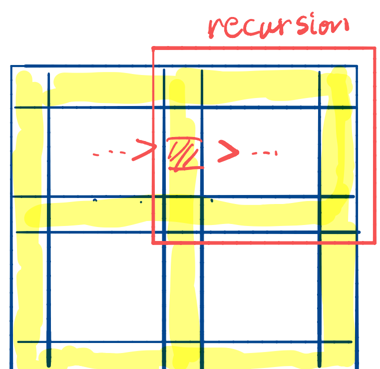
$A[mid-1] < A[mid] < A[mid+1]$ recurse into left half.

$$T(n) = T(n/2) + O(1) \implies T(n) = O(\log n)$$

Exmp (III) Local Minimum in 2D

Given a two-dimensional array $A[1 \,..\, n, 1 \,..\, n]$ of n^2 **distinct** numbers, and $i, j \in \{1, 2, \cdots, n\}$, we say that (i, j) is a local minimum of A, if
$A[i,j] < A[i,j-1], A[i,j] < A[i,j+1], A[i,j] < A[i-1,j]$ and $A[i,j] < A[i+1,j]$ (we assume that $A[i,j] = \infty$ if $i \in \{0, n+1\}$ or $j \in \{0, n+1\}$).
Suppose the array A is already stored in memory. Give an $O(n)$-time algorithm to find a local minimum of A.

▨ : the min element in the window frame.

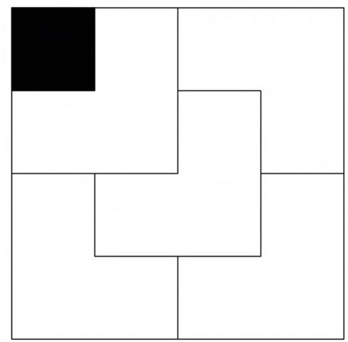


$$T(n) = T(n/2) + O(n)$$

$$\Rightarrow T(n) = O(\log n)$$

Exmp (IV) Chessboard tiling

Consider a $2^n \times 2^n$ chessboard with one arbitrary chosen square removed. Prove that any such chessboard can be tiled without gaps by L-shaped pieces, each composed of 3 squares. The following figure shows how to tile a 4×4 chessboard with the square on the left-top corner removed, using 5 L-shaped pieces.

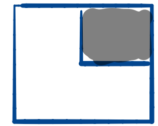


Input: n , (row, col)

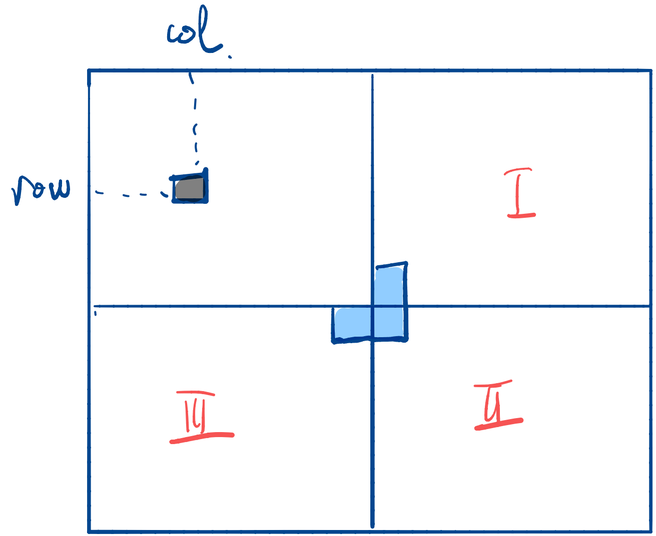
$\downarrow$

$(2^n \times 2^n)$-size board

Base case: $n = 1$



General case.



Key idea:

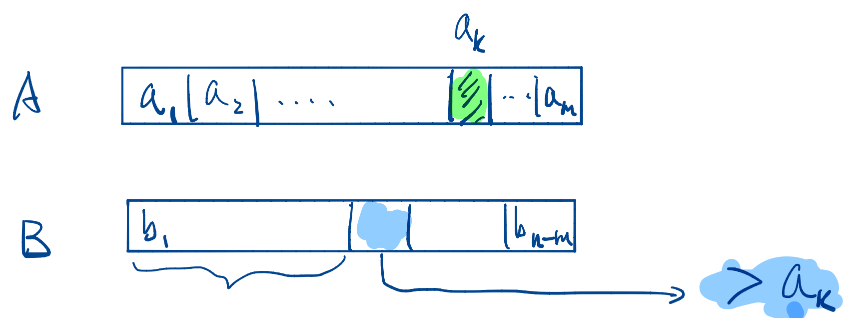
remove one L-shape in the center to make part I, II, III become smaller inst of chessboard tiling.

The recurse into the four parts.

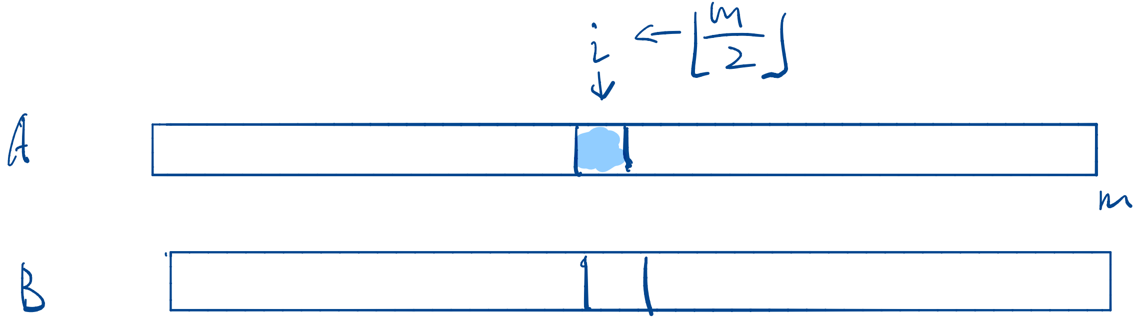
Exmp (V): Median of two arrays.

Given two sorted arrays A and B with total size n, you need to design and analyze an $O(\log n)$-time algorithm that outputs the median of the n numbers in A and B. You can assume n is odd and all the numbers are distinct.For example,

- Input: $A = [3, 5, 12, 18, 50]$,
- $B = [2, 7, 11, 30]$,
- Output: 11
- Explanation: the merged set is $[2, 3, 5, 7, 11, 12, 18, 30, 50]$



- $\left(\frac{n-1}{2} - k\right)$-many elements $< a_k$



$j = \frac{n-1}{2} - i$

$T(n) = O(\log n)$

First iter: $i \leftarrow \lfloor \frac{m}{2} \rfloor$, $j = \frac{n-1}{2} - i$

- check if $\begin{cases} B[j] < A[i] & \checkmark \\ B[j+1] > A[i] & \checkmark \end{cases}$ $\Rightarrow$ $A[i]$ is the true median
- If $B[j] > A[i]$? $\quad i \leftarrow \lfloor \frac{i+m}{2} \rfloor$
- If $B[j] < A[i]$ & $B[j+1] < A[i]$? $\quad i \leftarrow \lfloor \frac{i+0}{2} \rfloor$