Recitation (III): Divide-and-Conquer

Solving Recurrence

- $T(n) = 4T(n/3) + O(n)$  \quad \Rightarrow \quad T(n) = O(n^{\log_3 4})$
- $T(n) = 3T(n/3) + O(n)$  \quad \Rightarrow \quad T(n) = O(n \log n)$
- $T(n) = 4T(n/2) + O(n^{2\sqrt{n}})$  \quad \Rightarrow \quad T(n) = O(n^{2\sqrt{n}})$
- $T(n) = 8T(n/2) + O(n^3)$  \quad \Rightarrow \quad T(n) = O(n^3 \log n)$
Example (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 cannot multiply two big integers directly using a single operation.

Example:

\[
\begin{align*}
X &= 156978 \\
Y &= 723541 \\
\Rightarrow \quad X &= [8, 7, 9, 6, 5, 1] \\
Y &= [1, 4, 5, 3, 2, 7]
\end{align*}
\]

Equivalent to the polyrn multiplication problem:

\[
\begin{align*}
\mathbf{p}(a) &= 8 + 7a + 9a^2 + 6a^3 + 5a^4 + a^5 \\
\Rightarrow \quad X &= \mathbf{p}(10) \\
\mathbf{q}(a) &= 1 + 4a + 5a^2 + 3a^3 + 2a^4 + 1a^5 \\
\Rightarrow \quad Y &= \mathbf{q}(10)
\end{align*}
\]
Example (II). Local Minimum in 1D

Given an array $A[1..n]$ of $n$ distinct numbers, we say that some index $i \in \{1, 2, 3, \ldots, n\}$ 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 $O(\log n)$-time algorithm to find a local minimum of $A$.

<table>
<thead>
<tr>
<th>mid</th>
</tr>
</thead>
</table>


- recurse into right half.


- recurse into left half.

$T(n) = T(n/2) + O(1) \implies T(n) = O(\log n)$
Example (III) Local Minimum in 2D

Given a two-dimensional array $A[1 \ldots n, 1 \ldots n]$ of $n^2$ distinct numbers, and $i, j \in \{1, 2, \ldots, n\}$, 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 $A$. 

\[ T(n) = T(n/2) + O(n) \]

\[ \Rightarrow T(n) = O(\log n) \]
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 $4 \times 4$ chessboard with the square on the left-top corner removed, using 5 L-shaped pieces.

Input: $n$, (row, col) 
$(2^n \times 2^n)$-size board
Base case: $n = 1$

General case.

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

The recurse into the four parts.
Example (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 $[2, 3, 5, 7, 11, 12, 18, 30, 50]$

![Diagram showing the median calculation process]
First iter: \( i \leftarrow \left\lfloor \frac{m}{2} \right\rfloor, \ j = \frac{n-1}{2} - i \)

- if \( B[j] < A[i] \) \( \checkmark \) \( \Rightarrow A[i] \) is the true median
- if \( B[j] > A[i] \) \( \checkmark \)
- If \( B[j] > A[i] \)? \( i \leftarrow \left\lfloor \frac{i + m}{2} \right\rfloor \)
- If \( B[j] < A[i] \) \& \( B[j+1] < A[i] \)? \( i \leftarrow \left\lfloor \frac{i + 0}{2} \right\rfloor \)