CompSci 260P
Spring 2023 Lecture 16:
Divide and Conquer III
Integer Multiplication

- Given two \( n \)-bit integers \( X, Y \), compute \( X \cdot Y \)
- Example: What is \( 13 \cdot 11 \)?
What is a Computer Anyway?

Example: What is $13 \cdot 11$?
Why does al-Khwarizmi’s algorithm work?

Example: What is $13 \cdot 11$?

\[ X = 143 = \sqrt{100001111} \]
\[ X_{41} = 1000 = 8 \]
\[ 143 = 8 \cdot 2^{8/2} + 15 = 8 \cdot 16 + 15 \]
Starting D&C for Integer Multiplication

\[ X \times Y = (X_H \times 2^{n/2} + X_L) \times (Y_H \times 2^{n/2} + Y_L) \]

\[ = X_H \cdot Y_H \times 2^n + (X_H Y_L + X_L Y_H) \times 2^{n/2} + X_L Y_L \]

Algorithm Mult\((X, Y)\)

Create \(X_H, X_L, Y_H, Y_L\)

\[ A = \text{Mult}(X_H, Y_H) \]

\[ B = \text{Mult}(X_H, Y_L) \]

\[ C = \text{Mult}(X_L, Y_H) \]

\[ D = \text{Mult}(X_L, Y_L) \]

\[ \text{return } A \cdot 2^n + (B + C) \cdot 2^{n/2} + D \]

\[ T(n) = \Theta(T(\frac{n}{2}) + \Theta(n)) = \Theta(n^2) \]

\[ \Theta(n^{\log_2{3}}) \]

\[ E = \text{Mult}(X_H + X_L, Y_H + Y_L) \]
Do we really need to compute $X_H \cdot Y_L$?

\[
E = (X_H + X_L)(Y_H + Y_L) = X_H Y_H + X_L Y_H + X_H Y_L + X_L Y_L
\]
Example: What is $13 \cdot 11$?

<table>
<thead>
<tr>
<th>$X_H$</th>
<th>$X_L$</th>
<th>$Y_H$</th>
<th>$Y_L$</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>01</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
Minima Set Problem Statement

- We have a database of hotels.

- Each hotel has:
  - a proximity to the beach (x-coordinate)
  - a nightly room cost (y-coordinate)
  - Assume all coordinates distinct

- Want cheapest hotel closest to the beach
  - Might not be a unique hotel.
  - One might be closest, another cheapest.
  - Return the set that aren’t wrong.
    - Any where no other hotel is both cheaper and closer.
Minima Set Example

arrow = one reason
why elim
Minima Set Brute Force

Sort hotels along any dimension

\[
\text{for } i = 1 \rightarrow n - 1 \text{ do }
\]

\[
\text{for } j = i + 1 \rightarrow n \text{ do }
\]

\[
\text{if } A_i \text{ is cheaper and closer than } A_j \text{ then }
\]

Remove \( A_j \)

\[
\text{return } \text{All hotels that we did not remove}
\]

\[
\text{\quad This is } O(n^2).
\]
MinimaSet($S$)

if $n \leq 1$ then
    return $S$

$p \leftarrow$ median point in $S$ by $x$-coordinate
$L \leftarrow$ points less than $p$
$G \leftarrow$ points greater than or equal to $p$

$M_1 \leftarrow$ MinimaSet($L$)
$M_2 \leftarrow$ MinimaSet($G$)

\[ \text{return } M_1 \cup M_2? \]
Example revisited

From $M_1 \cup M_2$, which point(s) belong for sure?
Finding a correct recombine

MinimaSet(S)

if $n \leq 1$ then
    return $S$

$p \leftarrow$ median point in $S$ by $x$-coordinate
$L \leftarrow$ points less than $p$
$G \leftarrow$ points greater than or equal to $p$

$M_1 \leftarrow$ MinimaSet($L$)
$M_2 \leftarrow$ MinimaSet($G$)

$\Rightarrow$ return $M_1 \cup M_2$?

$\Rightarrow$ How do I recombine correctly?
Improved Recombine

\[ M_1 \leftarrow \text{MinimaSet}(L) \]
\[ M_2 \leftarrow \text{MinimaSet}(G) \]
\[ a = \text{min-cost in } M_1 \]

\[
\text{for each } a \in M_1 \text{ do }
\]
\[
\text{for each } b \in M_2 \text{ do }
\]
\[
\text{if } a \text{ better than } b \text{ then }
\]
\[
\text{remove } b \text{ from } M_2
\]

\[ T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n^2) \]

How can we improve the "recombine" step?

What is the resulting running time?