CS 577- Intro to Algorithms

Divide and Conquer
(Part 2)

Dieter van Melkebeek

September 15, 2020

Outline

Paradigm

1. Break up given instance into significantly smaller ones.
2. Recursively solve those.
3. Combine their solutions into one for the given instance.

Outline

Paradigm

1. Break up given instance into significantly smaller ones.
2. Recursively solve those.
3. Combine their solutions into one for the given instance.

Common pattern

Outline

Paradigm

1. Break up given instance into significantly smaller ones.
2. Recursively solve those.
3. Combine their solutions into one for the given instance.

Common pattern

- Sorting (Mergesort)
- Counting inversions

Outline

Paradigm

1. Break up given instance into significantly smaller ones.
2. Recursively solve those.
3. Combine their solutions into one for the given instance.

Common pattern

- Sorting (Mergesort)
- Counting inversions
- Closest pair of points in the plane (today)

Outline

Paradigm

1. Break up given instance into significantly smaller ones.
2. Recursively solve those.
3. Combine their solutions into one for the given instance.

Common pattern

- Sorting (Mergesort)
- Counting inversions
- Closest pair of points in the plane (today)

Other patterns

- Integer multiplication (today)

Outline

Paradigm

1. Break up given instance into significantly smaller ones.
2. Recursively solve those.
3. Combine their solutions into one for the given instance.

Common pattern

- Sorting (Mergesort)
- Counting inversions
- Closest pair of points in the plane (today)

Other patterns

- Integer multiplication (today)
- Selection (next time)

Recursion Tree Analysis of Common Pattern

Closest Pair of Points in the Plane

Closest Pair of Points in the Plane

Problem

Input: $(x_i, y_i) \in \mathbb{R}^2$ for $i \in [n]$

Closest Pair of Points in the Plane

Problem

Input: $(x_i, y_i) \in \mathbb{R}^2$ for $i \in [n]$

Output: $\delta \doteq \min\{\delta_{i,j} \text{ for } i,j \in [n] \text{ with } i \neq j\}$ where
$\delta_{i,j} \doteq \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

Closest Pair of Points in the Plane

Problem

Input: $(x_i, y_i) \in \mathbb{R}^2$ for $i \in [n]$

Output: $\delta \doteq \min\{\delta_{i,j} \text{ for } i, j \in [n] \text{ with } i \neq j\}$ where
$\delta_{i,j} \doteq \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

Algorithms

- Trivial: $O(n^2)$

Closest Pair of Points in the Plane

Problem

Input: $(x_i, y_i) \in \mathbb{R}^2$ for $i \in [n]$

Output: $\delta \doteq \min\{\delta_{i,j} \text{ for } i, j \in [n] \text{ with } i \neq j\}$ where

$\delta_{i,j} \doteq \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

Algorithms

- Trivial: $O(n^2)$
- Common D&C pattern: $O(n \log n)$

Closest Pair of Points in the Plane

Closest Crossing Pair in the Plane

Closest Pair of Points in the Plane

Pseudocode for recursive case

1. Find x^*, L, and R
2. Recursively compute δ_L and δ_R
3. $\delta^* \leftarrow \min(\delta_L, \delta_R)$
4. $M \leftarrow \{i \in [n] \text{ s.t. } x_i \in (x^* - \delta^*, x^* + \delta^*\}$
5. Sort M based on y-coordinate
6. $\delta_M \leftarrow \min\{\delta_{M[i],M[j]} \text{ for } i < j < i + 12\}$
7. Return $\min(\delta^*, \delta_M)$

Closest Pair of Points in the Plane

Pseudocode for recursive case

1. Find x^*, L, and R
2. Recursively compute δ_L and δ_R
3. $\delta^* \leftarrow \min(\delta_L, \delta_R)$
4. $M \leftarrow \{i \in [n] \text{ s.t. } x_i \in (x^* - \delta^*, x^* + \delta^*\}$
5. Sort M based on y-coordinate
6. $\delta_M \leftarrow \min\{\delta_{M[i],M[j]} \text{ for } i < j < i + 12\}$
7. Return $\min(\delta^*, \delta_M)$

Correctness

Closest Pair of Points in the Plane

Pseudocode for recursive case

1. Find x^*, L, and R
2. Recursively compute δ_L and δ_R
3. $\delta^* \leftarrow \min(\delta_L, \delta_R)$
4. $M \leftarrow \{i \in [n] \text{ s.t. } x_i \in (x^* - \delta^*, x^* + \delta^*\}$
5. Sort M based on y-coordinate
6. $\delta_M \leftarrow \min\{\delta_{M[i],M[j]} \text{ for } i < j < i + 12\}$
7. Return $\min(\delta^*, \delta_M)$

Correctness

Running time

Closest Pair of Points in the Plane

Pseudocode for recursive case

1. Find x^*, L, and R
2. Recursively compute δ_L and δ_R
3. $\delta^* \leftarrow \min(\delta_L, \delta_R)$
4. $M \leftarrow \{i \in [n] \text{ s.t. } x_i \in (x^* - \delta^*, x^* + \delta^*\}$
5. Sort M based on y-coordinate
6. $\delta_M \leftarrow \min\{\delta_{M[i],M[j]} \text{ for } i < j < i + 12\}$
7. Return $\min(\delta^*, \delta_M)$

Correctness

Running time

- Using local sorting: $O(n \log n)$ locally and $O(n(\log n)^2)$ overall

Closest Pair of Points in the Plane

Pseudocode for recursive case

1. Find x^*, L, and R
2. Recursively compute δ_L and δ_R
3. $\delta^* \leftarrow \min(\delta_L, \delta_R)$
4. $M \leftarrow \{i \in [n] \text{ s.t. } x_i \in (x^* - \delta^*, x^* + \delta^*\}$
5. Sort M based on y-coordinate
6. $\delta_M \leftarrow \min\{\delta_{M[i],M[j]} \text{ for } i < j < i + 12\}$
7. Return $\min(\delta^*, \delta_M)$

Correctness

Running time

- Using local sorting: $O(n \log n)$ locally and $O(n(\log n)^2)$ overall
- Using presorting: $O(n)$ locally and $O(n \log n)$ overall

Integer Multiplication

Integer Multiplication

Problem

Input: nonnegative integers a and b in binary notation

Output: product $a \times b$ in binary notation

Integer Multiplication

Problem

Input: nonnegative integers a and b in binary notation

Output: product $a \times b$ in binary notation

Grade school algorithm

$$
\begin{array}{rcccccccccc}
 & & & & & & 1 & 0 & 0 & 1 & 0 \\
\times & & & & & & 1 & 1 & 1 & 0 & 1 \\
\hline
 & & & & & & 1 & 0 & 0 & 1 & 0 \\
 & & & & 1 & 0 & 0 & 1 & 0 & & \\
 & & & 1 & 0 & 0 & 1 & 0 & & & \\
+ & & 1 & 0 & 0 & 1 & 0 & & & & \\
\hline
 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0
\end{array}
$$

Integer Multiplication

Problem

Input: nonnegative integers a and b in binary notation

Output: product $a \cdot b$ in binary notation

Algorithms

Integer Multiplication

Problem

Input: nonnegative integers a and b in binary notation

Output: product $a \cdot b$ in binary notation

Algorithms

- Grade school: $O(n^2)$

Integer Multiplication

Problem

Input: nonnegative integers a and b in binary notation

Output: product $a \cdot b$ in binary notation

Algorithms

- Grade school: $O(n^2)$
- D&C: $O(n^q)$ for some $q \in (1, 2)$

Integer Multiplication

Problem

Input: nonnegative integers a and b in binary notation

Output: product $a \cdot b$ in binary notation

Algorithms

- Grade school: $O(n^2)$
- D&C: $O(n^q)$ for some $q \in (1, 2)$
- Best known (2019): $O(n \log n)$

Integer Multiplication - D&C attempt

Integer Multiplication - recursion tree D&C attempt

Integer Multiplication - analysis D&C attempt

- Running time: $\left(\sum_{i=0}^{d-1} 2^i\right) \cdot c \cdot n + c' \cdot 4^d$ where $d = \log_2(n)$

Integer Multiplication - analysis D&C attempt

- Running time: $\left(\sum_{i=0}^{d-1} 2^i\right) \cdot c \cdot n + c' \cdot 4^d$ where $d = \log_2(n)$
- Geometric sum with ratio $r \neq 1$: $\sum_{i=0}^{d-1} r^i = \frac{r^d - 1}{r-1} = \Theta(r^d)$

Integer Multiplication - analysis D&C attempt

- Running time: $\left(\sum_{i=0}^{d-1} 2^i\right) \cdot c \cdot n + c' \cdot 4^d$ where $d = \log_2(n)$
- Geometric sum with ratio $r \neq 1$: $\sum_{i=0}^{d-1} r^i = \frac{r^d - 1}{r-1} = \Theta(r^d)$
- Geometric sum with $r = 2$: $\sum_{i=0}^{d-1} 2^i = 2^d - 1$

Integer Multiplication - analysis D&C attempt

- Running time: $\left(\sum_{i=0}^{d-1} 2^i\right) \cdot c \cdot n + c' \cdot 4^d$ where $d = \log_2(n)$
- Geometric sum with ratio $r \neq 1$: $\sum_{i=0}^{d-1} r^i = \frac{r^d - 1}{r-1} = \Theta(r^d)$
- Geometric sum with $r = 2$: $\sum_{i=0}^{d-1} 2^i = 2^d - 1$
- $2^d = n$ for $d = \log_2(n)$

Integer Multiplication - analysis D&C attempt

- Running time: $\left(\sum_{i=0}^{d-1} 2^i\right) \cdot c \cdot n + c' \cdot 4^d$ where $d = \log_2(n)$
- Geometric sum with ratio $r \neq 1$: $\sum_{i=0}^{d-1} r^i = \frac{r^d - 1}{r-1} = \Theta(r^d)$
- Geometric sum with $r = 2$: $\sum_{i=0}^{d-1} 2^i = 2^d - 1$
- $2^d = n$ for $d = \log_2(n)$
- $4^d = (2^2)^d = 2^{2d} = (2^d)^2 = n^2$

Integer Multiplication - analysis D&C attempt

- Running time: $\left(\sum_{i=0}^{d-1} 2^i\right) \cdot c \cdot n + c' \cdot 4^d$ where $d = \log_2(n)$
- Geometric sum with ratio $r \neq 1$: $\sum_{i=0}^{d-1} r^i = \frac{r^d - 1}{r-1} = \Theta(r^d)$
- Geometric sum with $r = 2$: $\sum_{i=0}^{d-1} 2^i = 2^d - 1$
- $2^d = n$ for $d = \log_2(n)$
- $4^d = (2^2)^d = 2^{2d} = (2^d)^2 = n^2$
- Running time: $O(n^2)$

Integer Multiplication - improved D&C

Integer Multiplication - improved D&C

- $a = a_L \cdot 2^{n/2} + a_R$
- $b = b_L \cdot 2^{n/2} + b_R$

Integer Multiplication - improved D&C

- $a = a_L \cdot 2^{n/2} + a_R$
- $b = b_L \cdot 2^{n/2} + b_R$
- $a \times b = (a_L \cdot 2^{n/2} + a_R) \times (b_L \cdot 2^{n/2} + b_R)$

Integer Multiplication - improved D&C

- $a = a_L \cdot 2^{n/2} + a_R$
- $b = b_L \cdot 2^{n/2} + b_R$
- $a \times b = (a_L \cdot 2^{n/2} + a_R) \times (b_L \cdot 2^{n/2} + b_R)$
- $a \times b =$
  $(a_L \times b_L) \cdot 2^n +$
  $(a_L \times b_R + a_R \times b_L) \cdot 2^{n/2} +$
  $(a_R \times b_R)$

Integer Multiplication - improved D&C

- $a = a_L \cdot 2^{n/2} + a_R$
- $b = b_L \cdot 2^{n/2} + b_R$
- $a \times b = (a_L \cdot 2^{n/2} + a_R) \times (b_L \cdot 2^{n/2} + b_R)$
- $a \times b =$
  $(a_L \times b_L) \cdot 2^n +$
  $(a_L \times b_R + a_R \times b_L) \cdot 2^{n/2} +$
  $(a_R \times b_R)$
- $a \times b =$
  $(a_L \times b_L) \cdot 2^n +$
  $[(a_L + a_R) \times (b_L + b_R) - (a_L \times b_L) - (a_R \times b_R)] \cdot 2^{n/2} +$
  $(a_R \times b_R)$

Integer Multiplication - recursion tree improved D&C

Integer Multiplication - analysis improved D&C

- Running time: $\left(\sum_{i=0}^{d-1}(\frac{3}{2})^i\right) \cdot c \cdot n + c' \cdot 3^d$ where $d = \log_2(n)$

Integer Multiplication - analysis improved D&C

- Running time: $\left(\sum_{i=0}^{d-1}(\frac{3}{2})^i\right) \cdot c \cdot n + c' \cdot 3^d$ where $d = \log_2(n)$
- Geometric sum with ratio r: $\sum_{i=0}^{d-1} r^i = \frac{r^d-1}{r-1} = \Theta(r^d)$

Integer Multiplication - analysis improved D&C

- Running time: $\left(\sum_{i=0}^{d-1}(\frac{3}{2})^i\right) \cdot c \cdot n + c' \cdot 3^d$ where $d = \log_2(n)$
- Geometric sum with ratio r: $\sum_{i=0}^{d-1} r^i = \frac{r^d - 1}{r - 1} = \Theta(r^d)$
- For $r = \frac{3}{2}$: $r^d = (\frac{3}{2})^d = \frac{3^d}{2^d}$

Integer Multiplication - analysis improved D&C

- Running time: $\left(\sum_{i=0}^{d-1}(\frac{3}{2})^i\right) \cdot c \cdot n + c' \cdot 3^d$ where $d = \log_2(n)$
- Geometric sum with ratio r: $\sum_{i=0}^{d-1} r^i = \frac{r^d-1}{r-1} = \Theta(r^d)$
- For $r = \frac{3}{2}$: $r^d = (\frac{3}{2})^d = \frac{3^d}{2^d}$
- For $d = \log_2(n)$
  - $2^d = n$

Integer Multiplication - analysis improved D&C

- Running time: $\left(\sum_{i=0}^{d-1}(\frac{3}{2})^i\right) \cdot c \cdot n + c' \cdot 3^d$ where $d = \log_2(n)$
- Geometric sum with ratio r: $\sum_{i=0}^{d-1} r^i = \frac{r^d-1}{r-1} = \Theta(r^d)$
- For $r = \frac{3}{2}$: $r^d = (\frac{3}{2})^d = \frac{3^d}{2^d}$
- For $d = \log_2(n)$
  - $2^d = n$
  - $3^d = 3^{\log_2(n)} = (2^{\log_2(3)})^{\log_2(n)} = (2^{\log_2(n)})^{\log_2(3)} = n^{\log_2(3)}$

Integer Multiplication - analysis improved D&C

- Running time: $\left(\sum_{i=0}^{d-1}(\frac{3}{2})^i\right) \cdot c \cdot n + c' \cdot 3^d$ where $d = \log_2(n)$
- Geometric sum with ratio r: $\sum_{i=0}^{d-1} r^i = \frac{r^d-1}{r-1} = \Theta(r^d)$
- For $r = \frac{3}{2}$: $r^d = (\frac{3}{2})^d = \frac{3^d}{2^d}$
- For $d = \log_2(n)$
  - $2^d = n$
  - $3^d = 3^{\log_2(n)} = (2^{\log_2(3)})^{\log_2(n)} = (2^{\log_2(n)})^{\log_2(3)} = n^{\log_2(3)}$
- Running time: $O(\frac{n^q}{n} \cdot c \cdot n + c' \cdot n^q) = O(n^q)$ where $q = \log_2(3) \approx 1.585$