Algorithms

Part c: Karatsuba's Algorithm

Ian Ludden

Learning Objectives

By the end of this lesson, you will be able to:

Learning Objectives

By the end of this lesson, you will be able to:

- Know the high-level structure of Karatsuba's algorithm and its big-O running time.

Learning Objectives

By the end of this lesson, you will be able to:

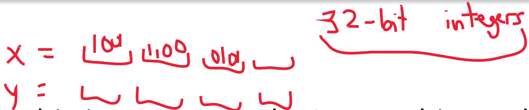
- Know the high-level structure of Karatsuba's algorithm and its big-O running time.
- Find a big-O solution for slightly harder recursive definitions, e.g., requiring use of the change of base formula.

Learning Objectives

By the end of this lesson, you will be able to:

- Know the high-level structure of Karatsuba's algorithm and its big-O running time.
- Find a big-O solution for slightly harder recursive definitions, e.g., requiring use of the change of base formula.
- Given a recursive algorithm (familiar or unfamiliar) express its running time as a recursive definition.

Multiplying Big Integers



Given big integers x and y ($n = 2m$ bits each), find product xy

Multiplying Big Integers

$X = x_1 x_0$ (bitstring concat.)

$140,729 = 140 \cdot 10^3 + 729$

Given big integers x and y ($n = 2m$ bits each), find product xy

Attempt 1: Divide and conquer! (maybe)

left shift

$$x = x_1 \cdot 2^m + x_0, \quad y = y_1 \cdot 2^m + y_0$$

Then,

Multiplying Big Integers

Given big integers x and y ($n = 2m$ bits each), find product xy

Attempt 1: Divide and conquer! (maybe)

$$x = x_1 \cdot 2^m + x_0, \quad y = y_1 \cdot 2^m + y_0$$

Then,

$$\begin{aligned} xy &= (x_1 \cdot 2^m + x_0)(y_1 \cdot 2^m + y_0) \\ &= x_1 y_1 \cdot 2^{2m} + (x_0 y_1 + x_1 y_0) \cdot 2^m + x_0 y_0 \\ &= A \cdot 2^{2m} + B \cdot 2^m + C. \end{aligned}$$

(1) (1) (2),(3) (4)

$T(1) = c$

$T(n) = 4T\left(\frac{n}{2}\right) + dn$ $\xrightarrow{\text{unrolling}}$ $O(n^2)$

Karatsuba's Algorithm

- Idea: Rearrange to eliminate one recursive call

Attempt 2: Divide and *conquer*!

Recall: $A = x_1 y_1$, $B = x_0 y_1 + x_1 y_0$, $C = x_0 y_0$.

Karatsuba's Algorithm

- Idea: Rearrange to eliminate one recursive call

Attempt 2: Divide and *conquer*!

Recall: $A = x_1 y_1$, $B = x_0 y_1 + x_1 y_0$, $C = x_0 y_0$.

Karatsuba's Algorithm

- Idea: Rearrange to eliminate one recursive call

Attempt 2: Divide and ***conquer***!

Recall: $A = x_1 y_1$, $B = x_0 y_1 + x_1 y_0$, $C = x_0 y_0$.
Observation: $B = (x_1 + x_0)(y_1 + y_0) - A - C$

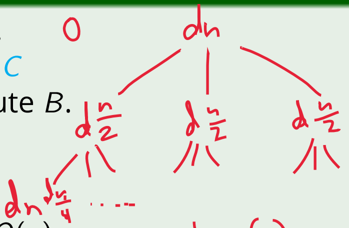
- Idea: Rearrange to eliminate one recursive call

Attempt 2: Divide and ***conquer***!

Recall: $A = x_1 y_1$, $B = x_0 y_1 + x_1 y_0$, $C = x_0 y_0$.
Observation: $B = (x_1 + x_0)(y_1 + y_0) - A - C$
Only one multiplication needed to compute B.

$$K(1) = c$$

$$K(n) = 3K(n/2) + O(n)$$



Recap: Learning Objectives

By the end of this lesson, you will be able to:

- Know the high-level structure of Karatsuba's algorithm and its big-O running time. $O(n^{\log_2 3})$
- Find a big-O solution for slightly harder recursive definitions, e.g., requiring use of the change of base formula.
- Given a recursive algorithm (familiar or unfamiliar) express its running time as a recursive definition.