COL106: Data Structures and Algorithms

Ragesh Jaiswal, IIT Delhi

Introduction

- How do Data Structures play a part in making computational tasks efficient?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

- Solution 1: Use long multiplication.
- What is the running time of the algorithm that uses long multiplication?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

- Solution 1: Use long multiplication.
- What is the running time of the algorithm that uses long multiplication? $O(n^2)$
- Is there a faster algorithm?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

- Solution 1: Algorithm using long multiplication with running time $O(n^2)$.
- Solution 2: (Assume n is a power of 2)
  - Write $A = A_L \cdot 2^{n/2} + A_R$ and $B = B_L \cdot 2^{n/2} + B_R$.
  - So, $A \cdot B = (A_L \cdot B_L) \cdot 2^n + (A_L \cdot B_R + A_R \cdot B_L) \cdot 2^{n/2} + (A_R \cdot B_R)$
  - Main Idea: Compute $(A_L \cdot B_L)$, $(A_R \cdot B_R)$, and $(A_R \cdot B_L)$, and $(A_L \cdot B_R)$ and combine these values.

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

- Solution 1: Algorithm using long multiplication with running time $O(n^2)$.
- Solution 2: (Assume n is a power of 2)
  - Write $A = A_L \cdot 2^{n/2} + A_R$ and $B = B_L \cdot 2^{n/2} + B_R$.
  - So, $A \cdot B = (A_L \cdot B_L) \cdot 2^n + (A_L \cdot B_R + A_R \cdot B_L) \cdot 2^{n/2} + (A_R \cdot B_R)$
  - Main Idea: Compute $(A_L \cdot B_L)$, $(A_R \cdot B_R)$, and $(A_R \cdot B_L)$, and $(A_L \cdot B_R)$ and combine these values.

Algorithm

`DivideAndConquer`(A, B)
- If $(|A| = |B| = 1)$ return$(A \cdot B)$
- Split A into A_L and A_R
- Split B into B_L and B_R
- $P \leftarrow$ `DivideAndConquer`(A_L, B_L)
- $Q \leftarrow$ `DivideAndConquer`(A_R, B_R)
- $R \leftarrow$ `DivideAndConquer`(A_L, B_R)
- $S \leftarrow$ `DivideAndConquer`(A_R, B_L)
- return(`Combine`(P, Q, R, S))

- What is the recurrence relation for the running time of the above algorithm?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

Algorithm

`DivideAndConquer`(A, B)
- If $(|A| = |B| = 1)$ return$(A \cdot B)$
- Split A into A_L and A_R
- Split B into B_L and B_R
- $P \leftarrow$ `DivideAndConquer`(A_L, B_L)
- $Q \leftarrow$ `DivideAndConquer`(A_R, B_R)
- $R \leftarrow$ `DivideAndConquer`(A_L, B_R)
- $S \leftarrow$ `DivideAndConquer`(A_R, B_L)
- return(`Combine`(P, Q, R, S))

- What is the recurrence relation for the running time of the above algorithm? $T(n) = 4 \cdot T(n/2) + O(n)$ for $n > 1$ and $T(1) = O(1)$.
- What is the solution to the above recurrence relation?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

Algorithm

`DivideAndConquer`(A, B)

- If $(|A| = |B| = 1)$ return$(A \cdot B)$
- Split A into A_L and A_R
- Split B into B_L and B_R
- $P \leftarrow$ `DivideAndConquer`(A_L, B_L)
- $Q \leftarrow$ `DivideAndConquer`(A_R, B_R)
- $R \leftarrow$ `DivideAndConquer`(A_L, B_R)
- $S \leftarrow$ `DivideAndConquer`(A_R, B_L)
- return(`Combine`(P, Q, R, S))

- What is the recurrence relation for the running time of the above algorithm? $T(n) = 4 \cdot T(n/2) + O(n)$ for $n > 1$ and $T(1) = O(1)$.
- What is the solution to the above recurrence relation? $T(n) = O(n^2)$.

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

- Solution 1: Algorithm using long multiplication with running time $O(n^2)$.
- Solution 2: Naïve Divide and Conquer with running time $O(n^2)$.
- Solution 3:
  - Write $A = A_L \cdot 2^{n/2} + A_R$ and $B = B_L \cdot 2^{n/2} + B_R$.
  - So, $A \cdot B = (A_L \cdot B_L) \cdot 2^n + (A_L \cdot B_R + A_R \cdot B_L) \cdot 2^{n/2} + (A_R \cdot B_R)$
  - Main Idea: Compute $(A_L \cdot B_L)$, $(A_R \cdot B_R)$, and $(A_L + B_L) \cdot (A_R + B_R) - (A_L \cdot B_L) - (A_R \cdot B_R)$.

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

Algorithm

`Karatsuba`(A, B)

- If $(|A| = |B| = 1)$ return$(A \cdot B)$
- Split A into A_L and A_R
- Split B into B_L and B_R
- $P \leftarrow$ `Karatsuba`(A_L, B_L)
- $Q \leftarrow$ `Karatsuba`(A_R, B_R)
- $R \leftarrow$ `Karatsuba`$(A_L + A_R, B_L + B_R)$
- return(`Combine`(P, Q, R))

- What is the recurrence relation for the running time of the above algorithm?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

Algorithm

`Karatsuba`(A, B)
- If $(|A| = |B| = 1)$ return$(A \cdot B)$
- Split A into A_L and A_R
- Split B into B_L and B_R
- $P \leftarrow$ `Karatsuba`(A_L, B_L)
- $Q \leftarrow$ `Karatsuba`(A_R, B_R)
- $R \leftarrow$ `Karatsuba`$(A_L + A_R, B_L + B_R)$
- return(`Combine`(P, Q, R))

- Recurrence relation: $T(n) \leq 3 \cdot T(n/2) + cn$; $T(1) \leq c$.
- What is the solution of this recurrence relation?

Problem

Multiplying two n-bit numbers: Given two n-bit numbers, A and B, Design an algorithm to output $A \cdot B$.

Algorithm

`Karatsuba`(A, B)

- If $(|A| = |B| = 1)$ return$(A \cdot B)$
- Split A into A_L and A_R
- Split B into B_L and B_R
- $P \leftarrow$ `Karatsuba`(A_L, B_L)
- $Q \leftarrow$ `Karatsuba`(A_R, B_R)
- $R \leftarrow$ `Karatsuba`$(A_L + A_R, B_L + B_R)$
- return(`Combine`(P, Q, R))

- Recurrence relation: $T(n) \leq 3 \cdot T(n/2) + cn$; $T(1) \leq c$.
- What is the solution of this recurrence relation?
  $T(n) \leq O(n^{\log_2 3})$

End