CS3000: Algorithms & Data
Jonathan Ullman

Lecture 3:
- Divide and Conquer: Karatsuba
- Solving Recurrences

Jan 13, 2020

Recap: Ask the Audience

- **"Big-Oh" Notation:** $f(n) = O\big(g(n)\big)$ if there exists $c \in (0, \infty)$ and $n_0 \in \mathbb{N}$ such that $f(n) \leq c \cdot g(n)$ for every $n \geq n_0$.

- Which of these statements are true?
  - $3n^2 + n = O(n^3)$ TRUE
  - $n^3 = O(n^2)$ FALSE
  - $\boxed{3n^2 + n = O(n^2)}$ TRUE
  - $\log_2(n^2) = O(\log_2(n))$

$$3n^2 + n = O(n^2)$$

$$\exists\, n_0, c \quad \text{s.t.} \quad \forall\, n \geq \underset{\substack{\shortparallel \\ 1}}{n_0} \quad f(n) \leq \underset{\substack{\shortparallel \\ 4}}{c} \cdot g(n)$$

$$3n^2 + n \leq 4n^2$$

$$3n^2 = O(n^2) \quad \text{(drop leading constant)}$$

$$n = O(n) = O(n^2) \quad \text{(bigger exponent)}$$

$$3n^2 + n = O(n^2) \quad \text{(add together terms)}$$

$$f(n) = \log_2(n^2) \qquad g(n) = \log_2(n)$$

$$\log_2(n^2) = O(\log_2(n))$$

$$\log_2(n^2) = 2 \cdot \log_2(n) = O(g(n))$$

$$3n^2 + n = O(n^3 + 100 n \log n)$$

Integer Multiplication: Karatsuba’s Algorithm

Addition

- Given n-digit numbers u, v output $u + v$

$$
\begin{array}{ccccc}
 & 1 & 2 & 3 & 4 \\
+ & 1 & 1 & 2 & 2 \\
\hline
= & 2 & 3 & 5 & 6
\end{array}
$$

$O(n)$ operations

Multiplication

- Given n-digit numbers u, v output $u \cdot v$

				1	2	3	4
			x	1	1	2	2
				2	4	6	8
+			2	4	6	8	0
+		1	2	3	4	0	0
+	1	2	3	4	0	0	0
	1	3	8	4	5	4	8

Running time of this alg is $\Theta(n^2)$

Divide and Conquer Multiplication

	1	**2**	**3**	**4**
x	**1**	**1**	**2**	**2**

$$\boldsymbol{u} = 10^2 \cdot \mathbf{12} + \mathbf{34}$$

$$\boldsymbol{v} = 10^2 \cdot \mathbf{11} + \mathbf{22}$$

	$\boldsymbol{a}$	$\boldsymbol{b}$
x	$\boldsymbol{c}$	$\boldsymbol{d}$

$$\boldsymbol{u} = 10^{n/2} \cdot \boldsymbol{a} + \boldsymbol{b}$$

$$\boldsymbol{v} = 10^{n/2} \cdot \boldsymbol{c} + \boldsymbol{d}$$

n is even

Divide and Conquer Multiplication

	$\boldsymbol{a}$	$\boldsymbol{b}$
x	$\boldsymbol{c}$	$\boldsymbol{d}$

$$\boldsymbol{u} = 10^{n/2} \cdot \boldsymbol{a} + \boldsymbol{b}$$

$$\boldsymbol{v} = 10^{n/2} \cdot \boldsymbol{c} + \boldsymbol{d}$$

$$\boldsymbol{u} \cdot \boldsymbol{v} = (10^{n/2} \cdot \boldsymbol{a} + \boldsymbol{b})(10^{n/2} \cdot \boldsymbol{c} + \boldsymbol{d})$$

$$= 10^{n} \cdot \boldsymbol{ac} + 10^{n/2} \cdot (\boldsymbol{ad} + \boldsymbol{bc}) + \boldsymbol{bd}$$

- Four $n/2$-digit mults, three n-digit adds
  - Multiplying by 10^n is “free” because it’s a shift
- Recurrence: $T(n) = 4T\left(\frac{n}{2}\right) + 3n$

Divide and Conquer Multiplication

$$T(n) = 4 \cdot T(n/2) + 3n$$
$$T(1) = 1$$

- **Claim:** $T(n) \geq n^2$

$$
\begin{aligned}
T(n) &= 4 \cdot T\left(\frac{n}{2}\right) + 3n \\
&\geq 4 \cdot \left(\frac{n}{2}\right)^2 + 3n \\
&= 4 \cdot \frac{n^2}{4} + 3 \\
&\geq n^2
\end{aligned}
$$

Karatsuba's Algorithm

	a	b
x	c	d

$$\boldsymbol{u} = 10^{n/2} \cdot \boldsymbol{a} + \boldsymbol{b}$$

$$\boldsymbol{v} = 10^{n/2} \cdot \boldsymbol{c} + \boldsymbol{d}$$

$$\boldsymbol{u} \cdot \boldsymbol{v} = 10^{n} \cdot \boldsymbol{ac} + 10^{n/2} \cdot (\boldsymbol{ad} + \boldsymbol{bc}) + \boldsymbol{bd}$$

- Key Identity
  - $(\boldsymbol{b} - \boldsymbol{a})(\boldsymbol{c} - \boldsymbol{d}) = \boldsymbol{ad} + \boldsymbol{bc} - \boldsymbol{ac} - \boldsymbol{bd}$

- Only three $n/2$-digit mults (plus some adds)!

Compute $b-a$, $c-d$

Compute ac, bd, $(b-a)(c-d)$

Karatsuba's Algorithm

Karatsuba(u,v,n):

  If (n = 1): Return $u \cdot v$ **// Base Case**

  Let $m \leftarrow \lceil n/2 \rceil$ **// Split**

  Write $u = 10^m \cdot a + b$**,** $v = 10^m \cdot c + d$

  Let e ← Karatsuba(a,c,m) **// Recurse**

      f ← Karatsuba(b,d,m)

      g ← Karatsuba(b-a,c-d,m)

  Return $10^{2m} \cdot e + 10^m \cdot (e + f + g) + f$ **// Merge**

Correctness of Karatsuba

- **Claim:** The algorithm **`Karatsuba`** is correct

Running Time of Karatsuba

```
Karatsuba(u,v,n):
   If (n = 1): Return u·v

   Let m ← ⌈n/2⌉
   Write u = 10^m a + b,  v = 10^m c + d

   Let e ← Karatsuba(a,c,m)
       f ← Karatsuba(b,d,m)
       g ← Karatsuba(b-a,c-d,m)

   Return 10^{2m} e + 10^m (e + f + g) + f
```

$O(n)$

$3 \times T(\frac{n}{2})$

$O(n)$

$$T(n) = 3 \times T\left(\frac{n}{2}\right) + Cn$$

Recursion Tree

$$T(n) = 3 \cdot T(n/2) + Cn$$
$$T(1) = C$$

Level	Problems	Work
0	n	Cn
1	$n/2$ $n/2$ $n/2$	$3 \times C\left(\frac{n}{2}\right) = C \times \left(\frac{3}{2}\right) \times n$
2	$n/4$	$9 \times C \times \frac{n}{4} = C \times \frac{9}{4} \times n$
$\vdots$		
i	$n/2^i \times 3^i$	$3^i \times C \times \frac{n}{2^i} = C \times \left(\frac{3}{2}\right)^i \times n$
$\vdots$		
$\log_2(n)$	1 …	$C \times \left(\frac{3}{2}\right)^{\log_2(n)} \times n$

$$C \times \frac{3^{\log_2(n)}}{2^{\log_2(n)}} \times n = C \times 3^{\log_2(n)} = C \times n^{\log_2(3)}$$

$$\sum_{i=0}^{\log_2(n)} C \times \left(\frac{3}{2}\right)^i \times n = C \times n \times \sum_{i=0}^{\log_2(n)} \left(\frac{3}{2}\right)^i$$

$$= C \times n \times C' \times \left(\frac{3}{2}\right)^{\log_2(n)}$$

$\log_2(3) \approx 1.59$

$$= \Theta\left(n^{\log_2(3)}\right)$$

Geometric Series

- Series $S = \sum_{i=0}^{\ell-1} r^i$

$S = 1 + r + r^2 + \ldots + r^{\ell-1}$

$rS = \quad r + r^2 + r^3 + \ldots + r^{\ell-1} + r^\ell$

$(1-r)S = 1 - r^\ell$

$S = \frac{1-r^\ell}{1-r} = \frac{r^\ell - 1}{r-1}$

- Solution:
  - $r \neq 1,\ S = \frac{1-r^\ell}{1-r} = \frac{r^\ell-1}{r-1}$
  - $r = 1,\ S = \ell$

$r > 1 :\ S = \Theta(r^\ell)$

$r < 1 :\ S = \Theta(1)$

$r = 1 :\ S = \Theta(\ell)$

Karatsuba Wrapup

- Multiply n digit numbers in $O(n^{1.59})$ time
  - Improves over naïve $O(n^2)$ time algorithm
  - **Fast Fourier Transform:** multiply in $\approx O(n \log n)$ time
- Divide-and-conquer approach
  - Uses a clever algebraic trick to split
  - **Key Fact:** adding is faster than multiplying
- Prove correctness via induction
- Analyze running time via recursion tree
  - $T(n) = 3T(n/2) + Cn$

Ask the Audience!

- $T(n) = 3T(n/2) + n^2$
- $T(1) = 1$

Level	Problems	Work
0	n	n^2
1	$n/2$ $n/2$ $n/2$	$3 \times \left(\frac{n}{2}\right)^2 = \frac{3}{4} \times n^2$
2		
$\vdots$		
i		$3^i \times \left(\frac{n}{2^i}\right)^2 = \left(\frac{3}{4}\right)^i \times n^2$
$\vdots$		
$\log_2(n)$		$\left(\frac{3}{4}\right)^{\log_2(n)} \times n^2$

$$n^2 \cdot \sum_{i=0}^{\log_2(n)} \left(\frac{3}{4}\right)^i = n^2 \cdot \left(\frac{1 - \left(\frac{3}{4}\right)^{\log_2(n)+1}}{1 - \frac{3}{4}} \right) \le n^2 \times 4$$

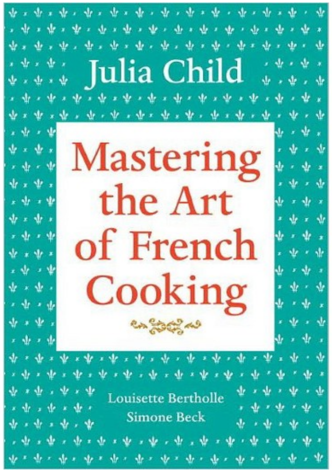
$$= O(n^2)$$

Solving Recurrences:
“The Master Theorem”

The "Master Theorem"

a, b, d independent of n

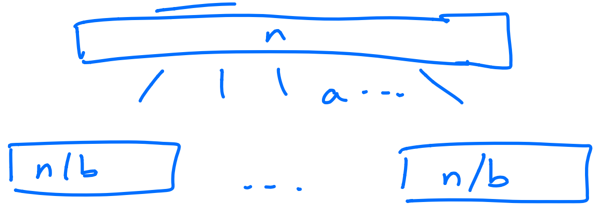
- Generic divide-and-conquer algorithm:
  - Split into $\boldsymbol{a}$ pieces of size $\frac{\boldsymbol{n}}{\boldsymbol{b}}$ and merge in time $O\left(n^{\boldsymbol{d}}\right)$
- Recipe for recurrences of the form:
  - $T(n) = \boldsymbol{a} \cdot T(n/\boldsymbol{b}) + Cn^{\boldsymbol{d}}$



Recursion Tree

- $T(n) = \boldsymbol{a}T(n/\boldsymbol{b}) + n^{\boldsymbol{d}}$
- $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^{\boldsymbol{d}}}\right) > 1$

Level | Problems | Work



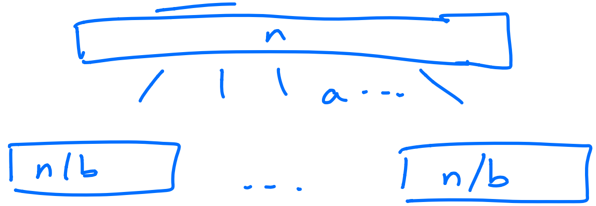
Level	Work
0	n^d
1	$a \times \left(\frac{n}{b}\right)^d = \left(\frac{a}{b^d}\right) \times n^d$
⋮	
i	$\left(\frac{a}{b^d}\right)^i \times n^d$
⋮	
$\log_b(n)$	$\left(\frac{a}{b^d}\right)^{\log_b(n)} \times n^d$

$$T(n) = n^d \times \sum_{i=0}^{\log_b(n)} \left(\frac{a}{b^d}\right)^i$$

Recursion Tree

- $T(n) = \boldsymbol{a}T(n/\boldsymbol{b}) + n^{\boldsymbol{d}}$
- $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^{\boldsymbol{d}}}\right) > 1$

Level	Problems	Work
0		n^d
1		$a \times \left(\frac{n}{b}\right)^d = \left(\frac{a}{b^d}\right) \times n^d$
$\vdots$		
i		$\left(\frac{a}{b^d}\right)^i \times n^d$
$\vdots$		
$\log_b(n)$		$\left(\frac{a}{b^d}\right)^{\log_b(n)} \times n^d$



$$T(n) = n^d \times \Theta\left(\left(\frac{a}{b^d}\right)^{\log_b(n)}\right) = \cancel{n^d} \times \Theta\left(\frac{a^{\log_b(n)}}{\cancel{n^d}}\right)$$

$$= \Theta\left(a^{\log_b(n)}\right) = \Theta\left(n^{\log_b(a)}\right)$$

Recursion Tree

- $T(n) = \boldsymbol{a}T(n/\boldsymbol{b}) + n^{\boldsymbol{d}}$
- $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^{d}}\right) = 1$

Mergesort : $a = 2$, $b = 2$, $d = 1$

$$T(n) = n^d \times \sum_{i=0}^{\log_b(n)} 1^i$$

$$= n^d \times \log_b(n) = \Theta(n^d \log(n))$$

Recursion Tree

- $T(n) = \boldsymbol{a}T(n/\boldsymbol{b}) + n^{\boldsymbol{d}}$
- $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^{\boldsymbol{d}}}\right) < 1$

Ex: $a = 3$

$b = 2$

$d = 2$

$$T(n) = n^d \times \sum_{i=0}^{\log_b(n)} \left(\frac{a}{b^d}\right)^i$$

$$\le n^d \times \sum_{i=0}^{\infty} \left(\frac{a}{b^d}\right)^i = n^d \times \frac{1}{1 - \frac{a}{b^d}} = \Theta\left(n^d\right)$$

The "Master Theorem"

- Recipe for recurrences of the form:
  - $T(n) = \boldsymbol{a} \cdot T(n/\boldsymbol{b}) + Cn^{\boldsymbol{d}}$
- Three cases:
  - $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^d}\right) > 1 : \; T(n) = \Theta\left(n^{\log_{\boldsymbol{b}} \boldsymbol{a}}\right)$
  - $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^d}\right) = 1 : \; T(n) = \Theta\left(n^{\boldsymbol{d}} \log n\right)$
  - $\left(\frac{\boldsymbol{a}}{\boldsymbol{b}^d}\right) < 1 : \; T(n) = \Theta\left(n^{\boldsymbol{d}}\right)$

Ask the Audience!

- Use the Master Theorem to Solve:
  - $T(n) = 16 \cdot T\left(\frac{n}{4}\right) + n^2$
  - $T(n) = 21 \cdot T\left(\frac{n}{5}\right) + n^2$
  - $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + 1$
  - $T(n) = 1 \cdot T\left(\frac{n}{2}\right) + 1$

The "Master Theorem"

- **Even More General:** all recurrences of the form
  - $T(n) = \boldsymbol{a} \cdot T(n/\boldsymbol{b}) + f(n)$
- Three cases:
  - $f(n) = O(n^{(\log_{\boldsymbol{b}} \boldsymbol{a})-\varepsilon})$:
    - $T(n) = \Theta\left(n^{\log_{\boldsymbol{b}} \boldsymbol{a}}\right)$
  - $f(n) = \Theta\left(n^{\log_{\boldsymbol{b}} \boldsymbol{a}}\right)$:
    - $T(n) = \Theta(f(n) \cdot \log n)$
  - $f(n) = \Omega(n^{(\log_{\boldsymbol{b}} a)+\varepsilon})$ **AND** $\boldsymbol{a}f\left(\frac{n}{\boldsymbol{b}}\right) \leq Cf(n)$ for $C < 1$
    - $T(n) = \Theta\left(f(n)\right)$