

CS 314 : Divide + Conquer

Note Title

2/5/2010

Announcements

- HW posted - due (written) next Friday
at start of class

Multiplying two numbers

Let's say we have an n -digit number

(Aside - how big can that number be?)

$$\leq 10^n$$

$$\begin{array}{r} \overset{1}{\overset{2}{\overset{3}{\overset{4}{\overset{5}{6}}}}} \ 3 \ 5 \ 2 \ 1 \ 9 \\ \times \quad \quad \quad \quad \quad 4 \\ \hline 2 \ 5 \ 4 \ 8 \ 8 \ 7 \ 6 \end{array}$$

if I give you a number $\leq n$, how many bits does it take to represent?

$$\log_2 n$$

$O(n)$ ^{1 digit} multiplications
 $O(n)$ additions

What about (2 n-digit) numbers?

$$\begin{array}{r} 625341 \\ \times 324950 \\ \hline 625341 \\ 3126705 \\ \hline 291 \end{array}$$

Diagram illustrating the multiplication process:

- The top number is 625341.
- The bottom number is 324950.
- Arrows point from each digit of the bottom number to its corresponding place value in the top number:
 - 3 points to the hundreds place in 625341.
 - 2 points to the tens place in 625341.
 - 4 points to the ones place in 625341.
 - 9 points to the thousands place in 625341.
 - 5 points to the ten-thousands place in 625341.
 - 0 points to the hundred-thousands place in 625341.
- A bracket on the right side groups the bottom number and the top number, labeled "n numbers to add together".
- Arrows point from the bottom number to the partial products:
 - An arrow from 3 points to 625341.
 - An arrow from 2 points to 625341.
 - An arrow from 4 points to 625341.
 - An arrow from 9 points to 625341.
 - An arrow from 5 points to 625341.
 - An arrow from 0 points to 625341.
- Arrows point from the partial products to the final result:
 - An arrow from 625341 points to 625341.
 - An arrow from 3126705 points to 291.
 - An arrow from 291 points to 291.

$O(n^2)$ 1-digit n

How would we code that algorithm?

Two nested for loops

$$O(n^2)$$

More efficient: Recursion!

$$\begin{aligned}
 & \overbrace{(10^m a + b)(10^m c + d)}^{\text{Original multiplication}} \\
 &= \underbrace{10^{2m} ac}_{\text{Red arrow}} + \underbrace{10^m(bc + ad)}_{\text{Red arrow}} + \underbrace{bd}_{\text{Red arrow}}
 \end{aligned}$$

Ex:

$$\begin{aligned}
 & 4563 \times 2729 \\
 &= (\underbrace{10^2 \cdot 45}_{\substack{\parallel \\ a}} + \underbrace{63}_{\substack{\parallel \\ b}})(\underbrace{10^2 \cdot 27}_{\substack{\parallel \\ c}} + \underbrace{29}_{\substack{\parallel \\ d}})
 \end{aligned}$$

Instead of calculating 1 big multiplication,
compute a·c, b·c, a·d, & b·d

Is this better? ←

Pseudocode

$x = 453269$

$a = \boxed{453}2\cancel{6}\cancel{9}$

$b = 269$

MULTIPLY(x, y, n):

```
if  $n = 1$ 
    return  $x \cdot y$ 
else
     $m \leftarrow \lceil n/2 \rceil$ 
     $a \leftarrow \lfloor x/10^m \rfloor$ ;  $b \leftarrow x \bmod 10^m$ 
     $d \leftarrow \lfloor y/10^m \rfloor$ ;  $c \leftarrow y \bmod 10^m$ 
     $e \leftarrow \text{MULTIPLY}(a, c, m)$ 
     $f \leftarrow \text{MULTIPLY}(b, d, m)$ 
     $g \leftarrow \text{MULTIPLY}(b, c, m)$ 
     $h \leftarrow \text{MULTIPLY}(a, d, m)$ 
    return  $10^{2m}e + 10^m(g + h) + f$ 
```

Running time: \downarrow to add $\#s$

$$R(n) = 4R\left(\frac{n}{2}\right) + O(n)$$
$$R(1) = 1$$

Exercise: use Master Thm

$$R(n) = O(n^2)$$

A trick: (Kanatsuba 1962)

$\underbrace{bc + ad}$ can be computed using only
 $\frac{\text{one}}{\uparrow}$ multiplication!

$$\rightarrow \overbrace{bc + ad}^{\uparrow} = \overbrace{ac + bd}^{\uparrow} - \overbrace{(a-b)(c-d)}^{\uparrow}$$

we already
compute these!

$$= \cancel{ac + bd} - \cancel{[ac - bc - ad + bd]}$$
$$= \cancel{bc + ad}$$

New + improved pseudo code:

```
FASTMULTIPLY( $x, y, n$ ):  
    if  $n = 1$   
        return  $x \cdot y$   
    else  
         $m \leftarrow \lceil n/2 \rceil$   
         $a \leftarrow \lfloor x/10^m \rfloor; b \leftarrow x \bmod 10^m$   
         $d \leftarrow \lfloor y/10^m \rfloor; c \leftarrow y \bmod 10^m$   
         $e \leftarrow \text{FASTMULTIPLY}(a, c, m)$   
         $f \leftarrow \text{FASTMULTIPLY}(b, d, m)$   
         $g \leftarrow \text{FASTMULTIPLY}(a - b, c - d, m)$   
        return  $10^{2m}e + 10^m(e + f - g) + f$ 
```



What's this running time?

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

$$T(1) = D(1)$$

↑
additions ← subtractions

$$T(n) = O\left(n^{\log_2 3}\right) = O\left(n^{1.58...}\right)$$

Exponentiation

How do we compute a^n ?

$$a^n = \underbrace{a \cdot a \cdot a \cdot a \cdots a}_n$$

$\alpha(n)$ multiplications

Naive algorithm :

```
SLOWPOWER(a, n):
    x ← a
    for i ← 2 to n
        x ← x · a
    return x
```

Running time? (# of multiplications)

$O(n)$

Faster idea:

$$a^n = a^{\lfloor \frac{n}{2} \rfloor} \cdot a^{\lceil \frac{n}{2} \rceil}$$

$$a^6 = a^3 \cdot a^3$$

$$a^{1001} = a^{500} \cdot a^{501}$$

Pseudocode:

```
FASTPOWER( $a, n$ ):  
    if  $n = 1$   
        return  $a$   
    else  
         $x \leftarrow \text{FASTPOWER}(a, \lfloor n/2 \rfloor)$  ←  
        if  $n$  is even  
            return  $x \cdot x$   
        else  
            return  $x \cdot x \cdot a$ 
```

Running time?

$$T(n) \leq T\left(\frac{n}{2}\right) + 2$$

$$T(1) = 1$$

$$\Rightarrow T(n) = O(\log_2 n)$$

$$= 2 \log_2 n$$

FastMultiply(x, x)