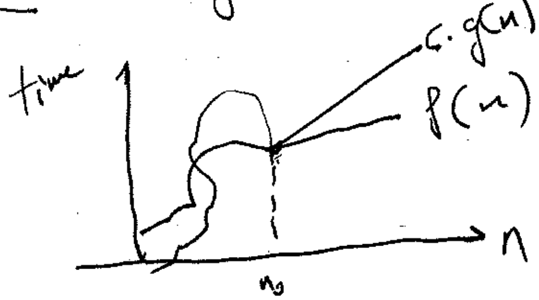
Lecture 3

Last time: Big O notation



$O(n) \approx \leq$

$\Omega(n) \approx \geq$

$\Theta(n) \approx =$

$o(n) \approx <$

$\omega(n) \approx >$

Today: basic arithmetic operations & time complexity } in binary

addition: $\underbrace{a}_{n\text{-bit}} + \underbrace{b}_{n\text{bit}} = \underbrace{c}$ size of this can not be more than $n+1$ bits.

multiplication: $a \times b = c$

$1101 \times 1011 =$

```
        1101   x
   *    1011
   ---------
        1101
       1101
      0000
 +   1101
 -----------
    10001111
```

multiply by 2 } by left shift by 1-bit

n times, each time n-bit #

$\Rightarrow O(n^2)$

```
mult2(x, y)
if x = 0 or y = 0 then return 0
if y = 1 return x
   z = mult2(x, ⌊y/2⌋)
   if y is even return 2.z
else y is odd return 2z+x
```

Analyze this Alg.

1- Time complexity

2- Correctness.

Time Complexity:

each call to mult(x,y) works on $\left\lfloor \frac{y}{2} \right\rfloor$ bits ≈ n-1 bits

$O(n)$ calls at most $\Rightarrow O(n)$.

$O(n) \cdot O(n) = O(n^2)$ bit complexity time

Correctness

Termination ✓

case I: y is even then we can write it as $y = 2 \cdot k$. for some k.

$$z = x \cdot \left\lfloor \frac{2k}{2} \right\rfloor = x \lfloor k \rfloor = xk$$

$$2z = 2xy \Rightarrow x \cdot 2k = xy \checkmark$$

case II: y is odd. write y as

$y = 2k+1$ for some k.

$$\left\lfloor \frac{y}{2} \right\rfloor = \left\lfloor \frac{2k+1}{2} \right\rfloor = \left\lfloor k + \tfrac{1}{2} \right\rfloor = k \checkmark$$

$$z = xk \Rightarrow 2z = 2xk + x = \dots = xy \quad \square$$

Mult3(x,y): reducing the # of bits by half.

$x = \boxed{x_L \mid x_R}$ (n-bit) $\qquad y = \boxed{}$ (n-bit)

ex: $\underline{1011}\,0110$

size of $|x_L| = \left\lceil \frac{n}{2} \right\rceil \quad |x_R| = \left\lfloor \frac{n}{2} \right\rfloor$ bits

$$x = 2^{n/2} x_L + x_R$$

in the example: $1011 \cdot 2^4 + 0110$

$$x \cdot y = \left(2^{n/2} x_L + x_R\right) \cdot \left(2^{n/2} y_L + y_R\right)$$

$$x \cdot y = 2^n x_L y_L + 2^{n/2}\left(x_L y_R + x_R y_L\right) + x_R y_R$$

observe that 4 pairs of n/2 bit numbers multiplied.

3 additions. $O(n)$

Let $T(n)$ be the running time of this alg. on 2 n-bit numbers

$$= 4T\left(\frac{n}{2}\right) + \theta(n)$$

$$= 4\left(4T\left(\frac{n}{4}\right) + \theta\left(\frac{n}{2}\right)\right) + \theta(n)$$

$$= 4^2\left(T\left(\frac{n}{4}\right)\right) + 4\theta\left(\frac{n}{2}\right) + \theta(n)$$

$$= 4^2\left(4T\left(\frac{n}{8}\right) + \theta\left(\frac{n}{4}\right)\right) + 4\theta\left(\frac{n}{2}\right) + \theta(n)$$

$$= 4^3 T\left(\frac{n}{2^3}\right) + 4^2\theta\left(\frac{n}{2^2}\right) + 4^1\theta\left(\frac{n}{2^1}\right) + 4^0\theta\left(\frac{n}{2^0}\right)$$

$$\Rightarrow = \sum_{i=0}^{\log n} \frac{4^i}{2^i}\theta(n)$$

$$= \sum_{i=0}^{\log n} 2^i \theta(n) = \theta(n)\sum_{i=0}^{\log n} 2^i$$

$$= \theta(n)\frac{2^{\log n+1} - 1}{2-1}$$

why! Geometric series

$$1 + r + r^2 + \dots r^k = \frac{r^{k+1}-1}{r-1}$$

$$= \theta(n)\ 2.\ 2^{\log n}$$

$$= 2.\ \theta(n) \cdot n = \theta(n^2)$$

Next: can we reduce the constant 4 i.e. 4-pairs of multiplication to say 3 $\Rightarrow$ decrease the time complexity

Due to C.F. Gauss

$$(a+bi)\cdot(c+di) = ac - bd + (bc+ad)i$$ 4 pairs of multiplications

$$bc + ad = \underbrace{(a+b)\cdot(c+d)}\ - \underbrace{ac} - \underbrace{bd}$$ } 3 pairs of multiplications

we will apply this to obtain a new multiplication alg. & bound

$$x\cdot y = (x_L + x_R)\cdot(y_L + y_R)$$

$$= \underbrace{x_L y_L}_{A} + (\underbrace{x_L y_R}_{C} + \underbrace{x_R y_L}_{D}) + \underbrace{x_R y_R}_{B.}$$ 4 pairs of multiplication

$$C, D = -A - B + (x_L + x_R)\cdot(y_L + y_R)$$

$$x\cdot y = 2^{2\lfloor\frac{n}{2}\rfloor}\underbrace{x_L y_L}_{A} + 2^{\lfloor n/2\rfloor}(\underbrace{x_L y_R + x_R y_L}_{C\ \ \ D}) + \underbrace{x_R y_R}_{B}$$ now we have 3 pairs of multiplication

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

$$T(n) = \sum_{i=0}^{\log n}\left(\frac{3}{2}\right)^i O(n)$$

$$= O(n)\sum_{i=0}^{\log n}\left(\frac{3}{2}\right)^i$$

$$= O(n)\,\frac{\left(\frac{3}{2}\right)^{\log n+1} - 1}{\left(\frac{3}{2}\right) - 1} = O\left(3^{\log n}\right) = O\left(n^{1.585}\right)$$

Modular Arithmetic

$r \mod_n \equiv x$

$0 - (n-1)$

$$\boxed{x = q \cdot n + r}$$

$7 = 2 \cdot 3 + 1 \Rightarrow \boxed{7 \mod_3} = 1$

$$\boxed{10^{10} \mod 7}$$

$10^{10} = k \cdot 7 + r$

practical example
RSA

msg $\Rightarrow$ # $\quad M^e \mod N = C$

$C^d \mod N = M$

$(x+y) \mod n = x \mod n + y \mod n$

$\frac{x}{y} \mod n = (x \cdot y^{-1}) \mod n$

multiplicative inverse of $y \mod n$

Definition: mult. inv. of $y \mod n$ is a number y^{-1} between 0 and $n-1$ such that

$y \cdot y^{-1} \equiv 1 \mod n.$