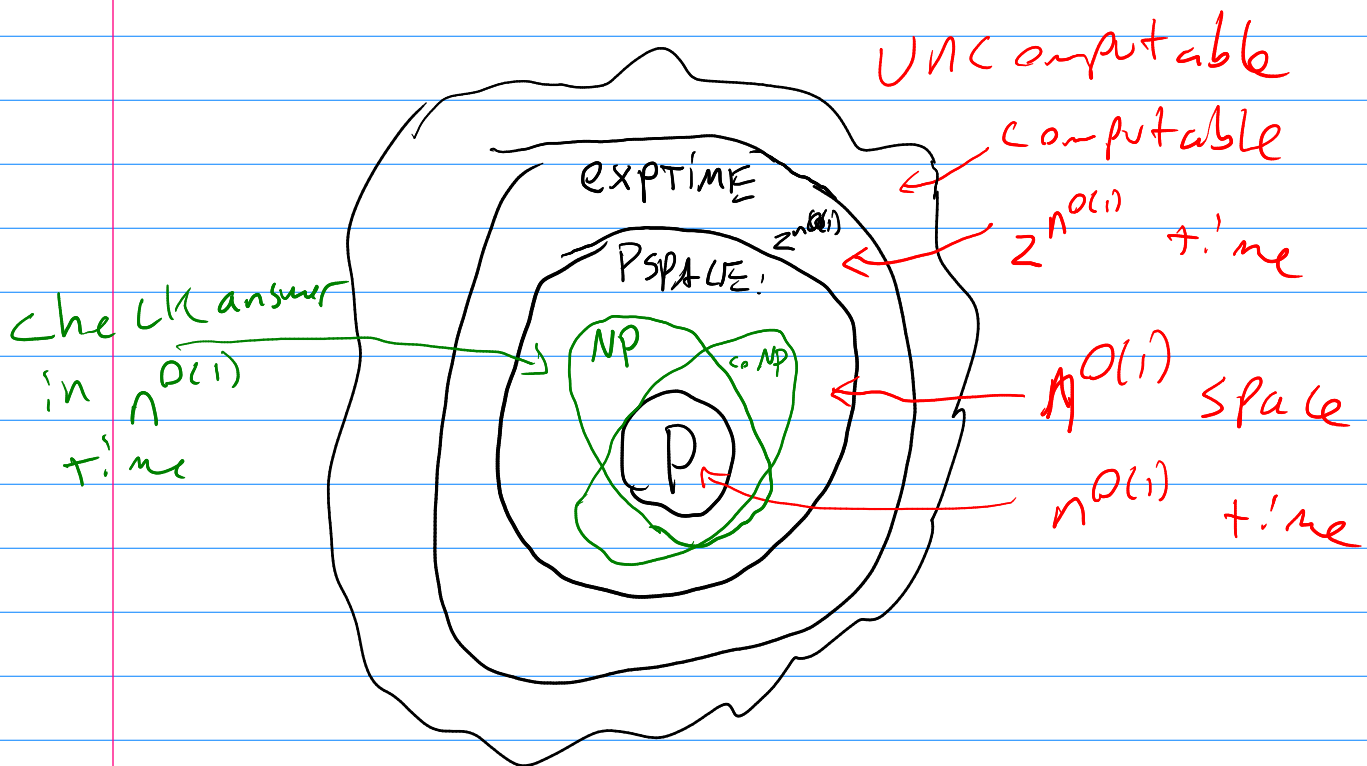


# Algorithms & Complexity

Notes 1

How quickly we can/can't solve problems

Complexity: Broad classification of problems



Algorithms: focuses within P.

Insertion Sort vs Merge Sort  
 $O(n^2)$  vs  $O(n \log n)$

Key topics:

- Dynamic Programming
- Graph Algorithms
- Linear Programming

# Example Algorithm

## Multiplication

$$\begin{array}{r} 331 \\ \times 388 \\ \hline 8 \\ 24 \\ 24 \end{array}$$

$$\begin{array}{r} 2648 \\ 2648 \\ 993 \\ \hline \end{array}$$

$$128428$$

$n$  intermediate products  
of length  $n$

How many steps for  $n$  digit multiplication?

First:  $n$  digit addition?  
single pass, right to left, constant time each  
 $\Rightarrow O(n)$  time.

multiplication:

multiply all pairs  
 $\Rightarrow O(n^2)$  time

Can we do better?

# Karatsuba multiplication

$$\underbrace{5124758048}_{A_1} \underbrace{9048}_{A_2} \times \underbrace{6172586337}_{B_1} \underbrace{6337}_{B_2}$$

$$A = 10^{n/2} \cdot A_1 + A_2$$

$$B = 10^{n/2} \cdot B_1 + B_2$$

$$AB = 10^n \cdot A_1 B_1 + 10^{n/2} (A_1 B_2 + B_1 A_2) + A_2 B_2$$

$n$  digit Product  
 = 4 products of  $\frac{n}{2}$  digits + 3 length- $O(n)$  additions

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

$$= O(n^2)$$

$$(A_1 + A_2)(B_1 + B_2) = A_1 B_1 + A_2 B_2 + A_1 B_2 + B_1 A_2$$

So suffices to compute 3 products of  $\frac{n}{2}$  digits:

$$A_1 B_1, A_2 B_2, \text{ and } (A_1 + A_2)(B_1 + B_2).$$

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

$$= O(n^{\log_2 3}) \approx O(n^{1.585})$$

$$\begin{aligned} & \log_2 n \\ & \downarrow \\ & 2 \log_2 \frac{n}{2} \\ & = n^2 \end{aligned}$$

$$\begin{aligned} & \log_2 n \\ & \downarrow \\ & 3 \log_2 \frac{n}{2} \\ & = n^{\log_2 3} \end{aligned}$$

# Fancier Results

Tom-Cook:

split 3 ways, do 5 multiplications

$$O(n^{\log_3 5}) \approx n^{1.465}$$

4 ways, do 7 multiplications

$$n^{\log_4 7} \approx n^{1.403}$$

K ways, do  $2K - 1$  multiplications

$$c(K) \cdot n^{\log_K (2K-1)} \quad \text{time}$$

↑  
exponential in K.

Schönhage - Strassen:  $O(n \log n \log \log n)$

via an FFT

Fürer, Harvey - van der Hoeven:  $O(n \log n)$

Simpler algorithms: better constants

Fancier algorithms: better asymptotics.