Lecture 19, 22 October 2026

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Programming and Data Structures with Python

Lecture 19, 22 Oct 2026

Complexity of algorithms

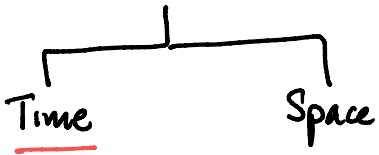
Algorithms + Data Structures = Programs

Niklaus Wirth

Pascal

↓

Efficiency / Cost / Resources



Time

FOCUS

Bias

Time is more important than space

Complexity of algorithms

Sort a 10 element list vs a 10^6 element list

Measure time as a function of input size

$n \longrightarrow T(n)$ time taken on input of size n

Complexity of algorithms

Intersection of 2 lists

Input size? $|l_1|, |l_2|$ — m, n

```
commonlist = []
for x in l1:          # m
    for y in l2:      # n
        if x == y:
            commonlist.append(x)
```

Time $(m \cdot n)$

Time n^2

Complexity of algorithms

Check that every SIM card has a valid Aadhaar number

```
validSIM = []
for x in SIM-aadhaar-list        m ≈ 10^9
    for y in Actual-aadhaar-list  n ≈ 10^9
        if x==y :  * Stop *                m·n
            validSIM.append(this SIM)
```

Complexity of algorithms

Time: $10^9 \times 10^9$ comparisons (or $0.5 \times 10^9 \times 10^9$)

10^{18} comparisons

CPU speed has plateaued since mid 2000's

Approx 10^9 operations/sec, max

Python $\simeq 10^7$ operations/sec

C++ $\simeq 10^8+$ operations/sec

Complexity of algorithms

Our code in Python

10^{18} ops, 10^7 ops/sec $\Longrightarrow$ 10^{11} secs

100 sec ~ 1.66 mins

1.7×10^9 mins

$1.7 \times 1.7 \times 10^7$ hours

3×10^7 hours

1×10^6 days

3000 years

Complexity of algorithms

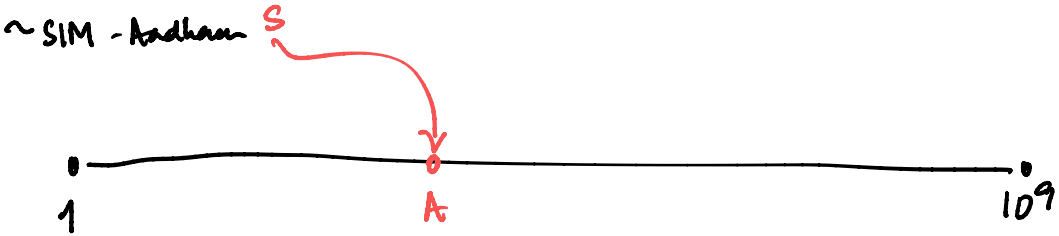
Data structures

We assumed nothing about how SIM & Aadhaar cards are organized

Arrange Aadhaar list in ascending order

Does this help?

Complexity of algorithms



$S = A$ ✓

$S < A$ - focus on $[1, A]$

$S > A$ - focus on $[A, 10^9]$

"Search space" reduces

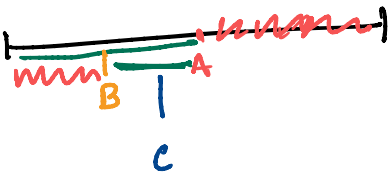
Complexity of algorithms

"Guaranteed" reduction of space

Choose midpoint [Assumes an array, random access]

$10^9 \rightarrow 10^9/2 \rightarrow 10^9/2^2$

$\rightarrow 10^9/2^3 \rightarrow \cdots \rightarrow 1$



$\frac{10^9}{2^k} = 1$ $\quad k = \log_2(10^9)$

$1 \rightarrow 2 \rightarrow 2^2 \cdots 10^9$?

$2^k = 10^9$

Complexity of algorithms

Binary Search

Halves search space each time

$N \longrightarrow \log_2(N)$ queries

What is $\log_2(10^9)$?

≈ 30

$2^{10} = 1024 \simeq 10^3$

$2^{20} = 2^{10} \cdot 2^{10} \simeq 10^6$

$2^{30} \simeq 10^9$

Complexity of algorithms

For each SIM Card $\sim 10^9$

Check if there is a matching Aadhaar Card

[Binary Search] ~ 30

3×10^{10} comparisons

$\frac{3 \times 10^{10}}{10^7}$ 3×10^3 secs

50 minutes

vs 3000 years

Complexity of algorithms

How to estimate time complexity?

- Function of input size n

$$n \longmapsto T(n)$$

"Ignore constant factors"

Complexity of algorithms

$1 + 2 + \cdots + n$

Inserting n elements into a sorted list, initially empty

$\frac{n(n+1)}{2} \rightarrow \frac{n^2}{2} + \frac{n}{2}$ vs n^2

How can we say these are both quadratic in n

$\frac{n^2}{2} + \frac{n}{2} \leq \frac{n^2}{2} + \frac{n^2}{2} = n^2$

Complexity of algorithms

f is similar "order of magnitude" as g

$\exists$ constant c . $\forall n$: $f(n) \leq c \cdot g(n)$

$5n^2 + 6n + 2$

$\leq 5n^2 + 6n^2 + 2n^2$

$\leq 13n^2$

$c = 13$

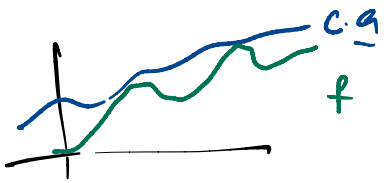
$5n^2 + 6n + 2 \leq 13 \cdot n^2$

Complexity of algorithms

$f = O(g)$ $\exists c.\ \forall n.\ f(n) \leq c \cdot g(n)$

Simple rule – keep the highest exponent

$3n^3 + 2n^2 + 6$ $4876\,n^2 + 5$



Complexity of algorithms

$n\sqrt{n}$

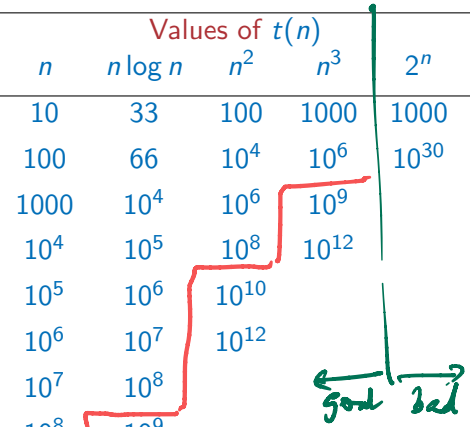
$\log_2 n \quad n \quad n\log_2 n \quad n\sqrt{n} \quad n^2 \; -- \; n^k \; -- \; 2^n$

Polynomial time

Exponential time

Orders of magnitude

Input size	Values of $t(n)$						
	$\log n$	n	$n \log n$	n^2	n^3	2^n	$n!$
10	3.3	10	33	100	1000	1000	10^6
100	6.6	100	66	10^4	10^6	10^{30}	10^{157}
1000	10	1000	10^4	10^6	10^9		
10^4	13	10^4	10^5	10^8	10^{12}		
10^5	17	10^5	10^6	10^{10}			
10^6	20	10^6	10^7	10^{12}			
10^7	23	10^7	10^8				
10^8	27	10^8	10^9				
10^9	30	10^9	10^{10}				
10^{10}	33	10^{10}	10^{11}				



Complexity of algorithms

$O(n^2)$ .. asymptotic complexity

Which input of size n?

Best case scenario ✗

✓ Worst case scenario – linear search, $\forall \exists$

Average case – How to measure average? Too hard

Complexity of algorithms

Simplex

