Order of Growth

- Abstraction to ease analysis

Best case of insertion sort

- Data is already sorted

$1 + 1 + 1 + 1 + \ldots + 1$ ($\sim n$ times)

$\Rightarrow n$ order of growth

n

Worst the case insertion sort

- Data is reverse sorted

7 5 3 2 1 + 1

5 7 3 2 1 + 2

3 5 7 2 1 + 3

2 3 5 7 1 + 4

1 2 3 5 7 5

$1+2+3+4+\cdots+n \approx \sum_{j=2}^{n} j = \frac{n(n+1)}{2}$

Look only at the leading term from mathematical exp

- Drop lower order terms
- Ignore the constant coefficient in the leading term

Worst Case

$\sim \frac{n(n+1)}{2} \Rightarrow \frac{n^2+n}{2} \Rightarrow \frac{n^2}{2} + \frac{1}{2}(n)$

$\Rightarrow an^2 + bn + c$ quadratic Equation

$\Rightarrow n^2$

order of growth

Average Case

why not

1) 9 7 5 3 1 | 7 9 11 13 19 ||

Normally because of nature of data to determine which is average case theoretically / mathematically will not be always possible.

2) Mostly avg case is closer to worst case only. And we really want tapper bound.

learn later on

⇒ probabilistic Analysis

Expected running time

⇒ randomized algorithm analy

pick data various location

design your analysis

Example:

$n + n \log_2 n$

$= n \log n$

1) Drop lower order term $- n$

$\Rightarrow n \log n$

2) Constant / Coeff

No constant

$\Rightarrow$ Order of growth $n \lg n$

Insertion sort was actually solved with 'incremental approach.

Divide and Conquer

merge sort

Divide: Splitting into two subarrays
A[p,q] & A[q+1, r]
q is halfway point

Conquer: Recursively sorting the two subarrays

Combine: Merging two sorted subarrays
to produce a single sorted A[p, r]

▷ Merge-sort (A, p, r)

if p < r then checking for Base

q ← ⌊(p+r)/2⌋ Divide

▷ merge-sort (A, p, q) Conquer

▷ merge-sort (A, q+1, r) Conquer

□ merge (A, p, q, r) Combine (iterative)

Initial call merge-sort (A, 1, n)

- Solving recurrences
- Master method