Lecture 2

Insertion sort

Insertion-sort (A, n):   (Info: Arrays are 1-indexed)

```
for i = 2 to n
    key = A[i]
    j = i-1
    while j > 0 and A[j] > key
        A[j+1] = A[j]
        j = j-1
    A[j+1] = key
```

$$\begin{array}{cccccc} j & & j & & i & \\ \boxed{1} & \boxed{4} & \boxed{5} & \boxed{6} & \boxed{2} & \boxed{3} \\ & & 4 & 5 & 6 & \end{array}$$

$i = 5$

$key = 2$

How to measure run time of an algo?

- Code it and check the runtime. X
- It's the number of basic operations an algo performs as a function of the input size.

$A[i] = 2$, $a = b + c$, $a > key$, etc.

Insertion-sort (A, n): → for (i=2; i<=n; i++)

	# basic op.	times
for i = 2 to n	2	n
key = A[i]	1	$n-1$
j = i-1	1	$n-1$
while j > 0 and A[j] > key	2	$\sum_{i=2}^{n} t_i$
A[j+1] = A[j]	1	$\sum_{i=2}^{n} (t_i - 1)$
j = j-1	1	"
A[j+1] = key	1	$n-1$

4	5	6	3	2	1

$i=4$, $t_4 = 4$, $t_i = i$

t_i = # of times while loop is tested for that value of i.

Let $T(n)$ = # of basic-ops of Ins. Sort. on input length n.

$$T(n) = 2n + (n-1) + (n-1) + 2 \cdot \sum_{i=2}^{n} t_i + 2 \cdot \sum_{i=2}^{n} (t_i - 1) + n - 1$$

$$= 2n + 3(n-1) + 2 \sum_{i=2}^{n} t_i + 2 \sum_{i=2}^{n} (t_i - 1)$$

Best Case: when array is already sorted.

1	2	3	4	5	6

$t_i = 1$

$$= 2n + 3 \cdot (n-1) + 2 \cdot \sum_{i=2}^{n} 1 + 2 \cdot \sum_{i=2}^{n} (1-1)$$

$$= 2n + 3 \cdot (n-1) + 2 \cdot (n-1) = an + b, \text{ where } a, b \text{ are some constant.}$$

$$= 7n - 5 = O(n)$$

Why only the leading term matters?

It predicts the comparitive perf. of diff. algos.

Considers two algos:

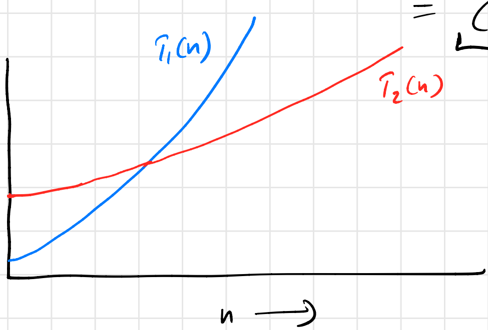
$$\underline{n^2 > c.\, n^{1.9}}$$

Algo A's runtime, $T_1(n) = n^2 + 100$

Algo B's runtime, $T_2(n) = 10000.\, n^{1.9} + 4000\, n^{1.5} + 9004 + 10^7$

$$= \underbrace{C.\, n^{1.9}}$$

$$\underline{\underline{n = 100^{100^{100}}}}$$



Algo B outperforms A for large values of n.

Worst case:

$$T(n) = 2n + 3.(n-1) + 2.\sum_{i=2}^{n} i + 2.\sum_{i=2}^{n} (i-1)$$

$$= 2n + 3.(n-1) + 2.\left(\frac{n.(n+1)}{2} - 1\right) + 2.\frac{(n-1).n}{2}$$

$$= \underbrace{a'n^2 + b'.n + c'}$$

$$= \underline{\underline{O(n^2)}}$$

$$\sum_{i=2}^{n} i = \underbrace{2+3+4+\cdots+n}$$

$$1+2+\cdots+n = \frac{n.(n+1)}{2}$$

Divide and Conquer method:

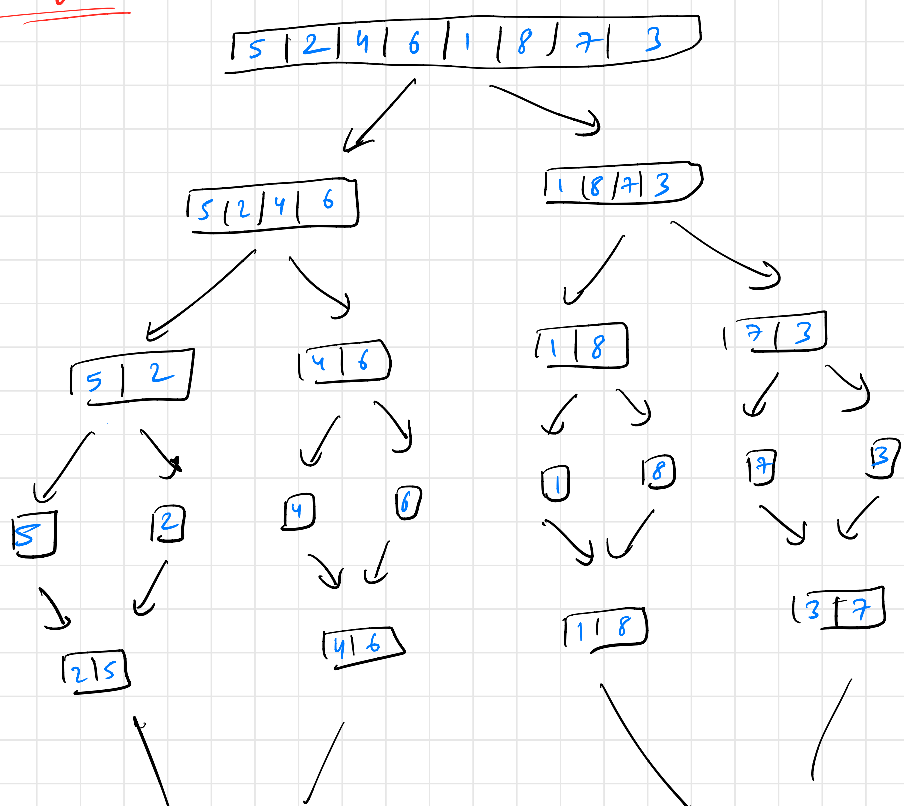
Three char. steps:

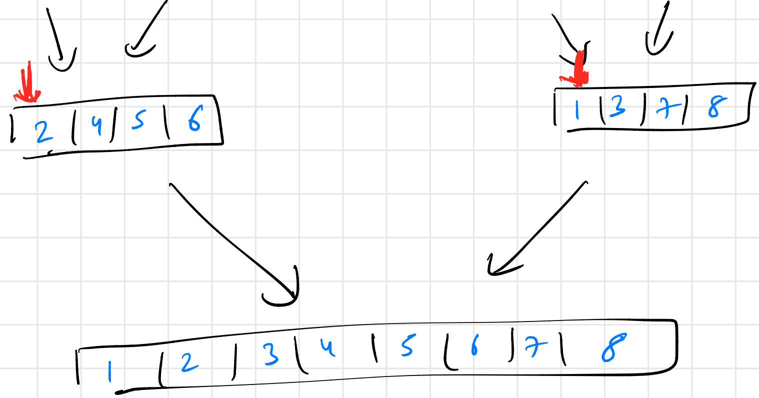
Divide: Break the problem into one or more smaller subproblems

Conquer: Solve the subproblems recursively.

Combine: Use solutions to subproblems to create sol. for the problem.

Merge Sort:



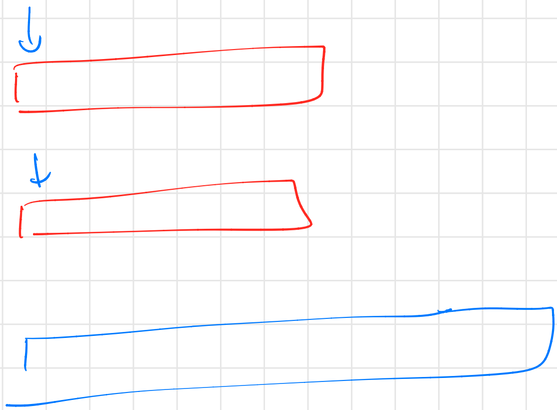


5

Cn^2

$c' n \log n$

100



$n = 10^8$

$n = 10^8$

Intro. to algorithms CLRS