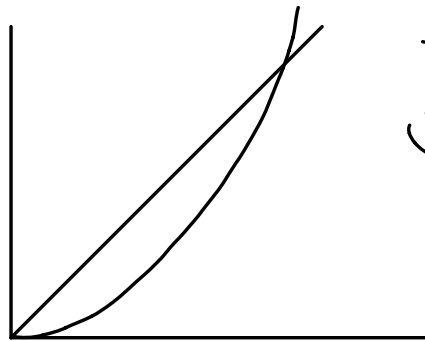


Lecture 18

Comparing Algorithms

- compare running time
- we will see a "rough measure" that characterizes how fast the function grows (rate of growth)
- compare functions for LARGE values of n , that is, compare functions in the limit (ASYMPTOTICALLY)

Ex) choice of linear-time program and quadratic-time program



$$f(n) = 100n$$
$$g(n) = 2n^2$$

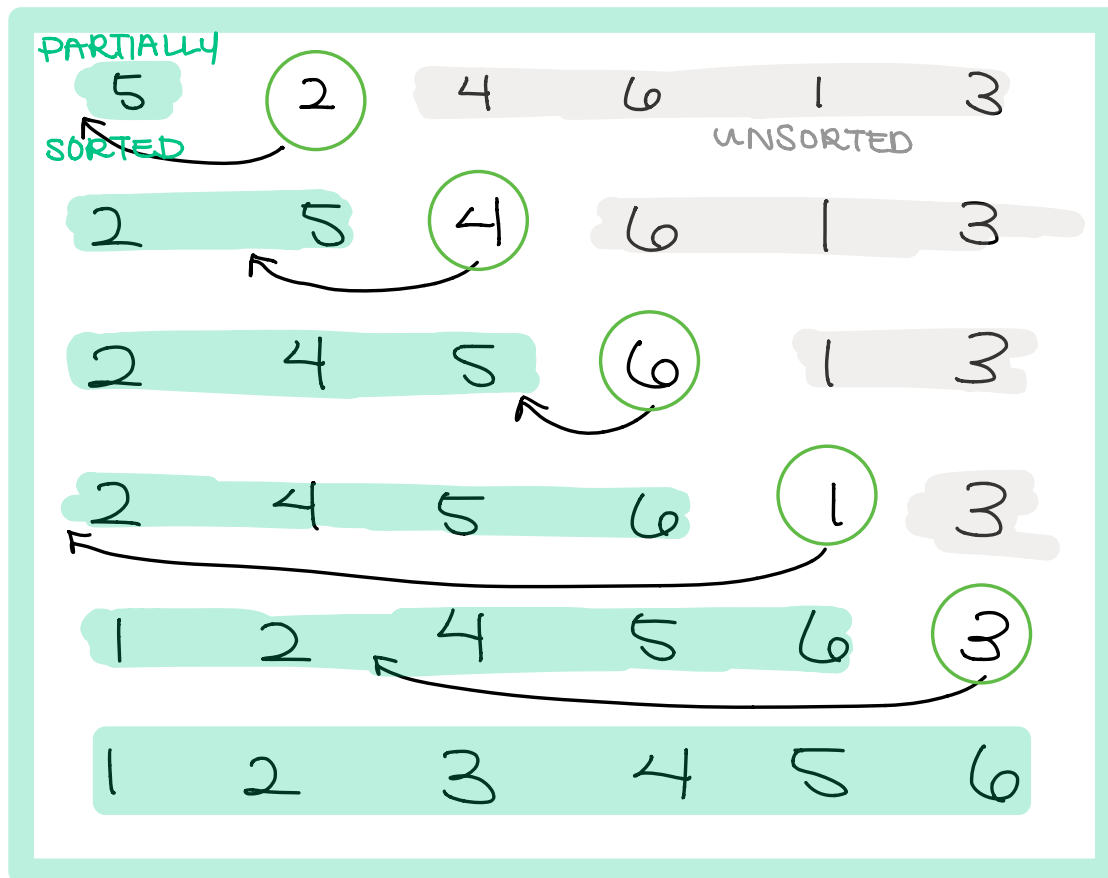
Sorting Algorithm

- puts elements of a list in a certain order
- Ex. Insertion Sort

- how most people sort a deck of cards
- pick the next card (to be sorted),

make room for it by shifting sorted items,
insert it into the correct location

Illustration : Input 5 2 4 6 1 3



Algorithm

(1) for $i = 2$ to n do

(2) $next = A[i]$

(3) $j = i - 1$

(4) while ($j > 0$ and $A[j] > next$)

(5) $A[j + 1] = A[j]$

(6) $j = j - 1$;

(7) $A[i] = next$;

$$f(n) = \Theta(n^2)$$

Cost

times

$t_i = \#$ of times while statement is executed at iteration i

C_1

n

C_2

$n-1$

C_3

$n-1$

C_4

BEST CASE

WORST CASE

1	$\sum_{i=2}^n$	t_i
0		t_{i-1}
0		t_{i-1}

C_5

C_6

C_7

$n-1$

Total Cost

$$f(n) = C_1 n + C_2 (n-1) + C_3 (n-1) + C_4 \sum_{i=2}^n t_i + C_5 \sum_{i=2}^n t_{i-1} + C_6 \sum_{i=2}^n t_{i-1} + C_7 (n-1)$$

$$\sum_{k=1}^n k = \frac{n(n+1)}{2}$$

Best Case Analysis

- array is already sorted
 $A[j] \leq$ next upon the first time the while loop is run. Then $t_i = 1$ for each $i = 2, \dots, n$

$$f(n) = (C_1 + C_2 + C_3 + C_4 + C_7)n - (C_2 + C_3 + C_4 + C_7)$$

$$= k_1 n + k_2$$

$f(n) = - (n)$ rate of growth is

Worst Case Analysis

Array is in reverse order

$A[j] > \text{next}$ in while loop

Worst case analysis: The array is sorted in reverse order. Always $A[j] > \text{next}$ in while-loop test. Then we have to compare next with all elements to the left of the i^{th} position. That is, we compare with $i-1$ element. So, $t_i = i$. It follows that

$$f(n) = C_1 n + C_2(n-1) + C_3(n-1) + C_4 \sum_{i=2}^n i + C_5 \sum_{i=2}^n (i-1) + C_6 \sum_{i=2}^n (i-1) + C_7(n-1)$$

$\sum_{i=2}^n i = \frac{n(n+1)}{2} - 1$
 $\sum_{i=2}^n (i-1) = \sum_{k=1}^{n-1} k = \frac{n(n-1)}{2}$

$$f(n) = C_1 n + C_2(n-1) + C_3(n-1) + C_4 \left(\frac{n(n+1)}{2} - 1 \right) + C_5 \frac{n(n-1)}{2} + C_6 \frac{n(n-1)}{2} + C_7(n-1)$$

$$= k_1 n^2 + k_2 n + k_3$$