CSE525 Lec1: Recursion

•••

Debajyoti Bera (M21)

~~https://sites.google.com/a/iiitd.ac.in/cse525-m19~~

Insertion Sort

54	26	93	17	77	31	44	55	20	Assume 54 is a sorted list of 1 item
26	54	93	17	77	31	44	55	20	inserted 26
26	54	93	17	77	31	44	55	20	inserted 93
17	26	54	93	77	31	44	55	20	inserted 17
17	26	54	77	93	31	44	55	20	inserted 77
17	26	31	54	77	93	44	55	20	inserted 31
17	26	31	44	54	77	93	55	20	inserted 44
17	26	31	44	54	55	77	93	20	inserted 55
17	20	26	31	44	54	55	77	93	inserted 20

Fix(i) : Assuming A[1...i-1] is sorted, sort A[1...i]. Fix(5) when A= [3,4,1,2,5] ?

Fix(5) when A= [1,3,4,5,2] ✓

→IS(i) : Sorts A[1...i] using insertion sort.

A[1]... A[i] will be sorted.

InsertionSort(): ✓

~~IS(n)~~ IS(1) IS(2) ... IS(i-1) | IS(i) ... IS(n)

IS(i): Return if i=0 or i>n

→ ~~Return if i=n+1~~

Fix[illegible]t(i)

~~IS(i+1)~~

← A[1,...,i-1] may be unsorted

IS(i): if i≤1 or i>n return

IS(i-1) // A[1...i-1] will be sorted

Fix(i)

InsertionSort(): IS(n)

A=([3,4,1,2,5,7])

IS(5)

→([1,2,3,4,5,7])

Proof of Correctness

```
Fix(i) : Assuming A[1...i-1] is
sorted, sort A[1...i].

IS(i):
      1. Return        if       i=0
      2.                      IS(i-1)
3. Fix(i)
```

Claim: IS(i) sorts A[1..i] in place.
Proof by induction on i.
Base case: When i=0, the claim is trivially true since

- A[1...0] is an empty array.
- IS(0) returns immediately

Induction hypothesis: Assume that IS(i-1) sorts A[1...(i-1)] in place.

Induction statement: *To prove that IS(i) sorts A[1..i] in place*, for i > 0.

IS(i) first calls IS(~~A,~~i-1). By IH, after this, A[1...(i-1)] is sorted.

IS(i) then calls Fix(i~~-1~~) which inserts A[i] inside the already sorted array A[1...(i-1)] in its right place in the sorted order. So after this call A[1...i] will be sorted.

Analysis of Insertion Sort

Assume that Fix(j) takes O(j) steps.

Let T(k) denote the time complexity of IS(k).

Write a recursive expression for T(k). $T(k) = T(k-1) + O(k) \quad = O(k^2)$

Exercise

```
Fix(j) : Assume A[1...j] is sorted, except its last element.
Place the min element of A[1...j] in the right place inside A[1...j].
```

Write a recursive algorithm for ~~FixPosLast(A)~~. Fix(j) .

(Optional) Prove that your algorithm is correct using induction.

} Fix(j)

Can you speed up Fix(j) ? Either asymptotically or according to clock time.

Is this correct?

```
IS(i):
  Return if i=0
  IS(i-1)
  FixPosLast(i)
```

Recursive Sort by Dividing at the Middle

1. Pre-process array.
2. Assume **left-half** is correctly sorted.
3. Assume **right-half** is correctly sorted.
4. Construct sorted version of the entire array.

7	11	6	55	98	45	16	96	46

Merge-sort

1. ? no pre-processing

2,3. ✓

4. ? Merge()

Time-complexity ?

Recurrence for Worst case compl.

$T(n) = 2T(n/2) + O(n)$

Quick-sort

1. ? partition, pivot

2,3. ✓

4. ? no-op

(Worst) time-complexity ?

Let T(n) denote the worst case time complexity of QS on n elements

Worst-case time complexity of QS

$$T(n) = T(n-1) + O(n)$$

k n-k

```
QuickSort(A[1..n]):
  if (n > 1)
      Choose a pivot element A[p]
      r ← Partition(A, p)
      QuickSort(A[1..r − 1])
      QuickSort(A[r + 1..n])
```

$$T(n) = \max_{k=1\ldots n} \left\{ T(k) + T(n-k) + \underline{O(n)} \right\}$$

$$= \max_{k=1\ldots n} \left\{ T(k) + T(n-k) \right\} + O(n)$$

doesn't depend on k

What is the recurrence for T(n) in terms of r ?

What is the recurrence for the worst-case expression for T(n) ?