Lect Note 2

A1

IIIT BHUBANESWAR

Divide and Conquer Algorithms: Binary Search

January, 2018

Instructor: Ajaya Kumar Dash.

Problem Statement:

We have to determine whether a given element x is present in a sorted non-decreasing array A or not.

If x is present in the array A, binary search will provide us the index of x in the input array A. On the other hand, if the element is not present in the array A, then the return result will be -1.

$T(n) \Leftarrow$

```
BinarySearch (A, x, p, r)
if p == r                                   → 1 time step
    if x == A[p]                            → 1 time step
        return p                            → 1 time step  (or)
    else
        return −1                           → 1 time step
else
    q = ⌊(p+r)/2⌋                           → 1 timestep
    if x ≤ A[q]                             → 1 timestep
        return BinarySearch (A, x, p, q)    ≈ → T(n/2)  (or)
    else
        return BinarySearch (A, x, q+1, r)  ≈ T(n/2)
```

Base case (For one element)

D & C Steps

1 division

2 conquer

Complexity Analysis of Binary search (worst case)

$T(1) = 1+1+1 = 3$-time steps (we can consider it as constant 'd')

$T(n) = \cancel{\text{Time for Division}} + \cancel{\text{Time for conquer}} + \cancel{\text{Time for combine}}$

1 (consider some const 'c')

$T\left(\frac{n}{2}\right)$

0

NB: No combine step in Binary Search

$\Rightarrow T(n) = T\left(\frac{n}{2}\right) + C$

using substitution method

$\Rightarrow T(n) = T\left(\frac{n}{2}\right) + c$

$= \left[T\left(\frac{n}{2^2}\right) + c\right] + c$

$= T\left(\frac{n}{2^2}\right) + 2c$

$= \left[T\left(\frac{n}{2^3}\right) + c\right] + 2c$

$= T\left(\frac{n}{2^3}\right) + 3c$

$\vdots$

$= T\left(\frac{n}{2^k}\right) + kc$

put $\frac{n}{2^k} = 1 \Rightarrow 2^k = n \Rightarrow k = \log_2 n$

$\Rightarrow T(n) = \underbrace{T(1)}_{d} + (\log_2 n)c$

$\Rightarrow T(n) = c\log_2 n + d$

$\Rightarrow T(n) \leq \underbrace{(c+d)}_{c'}\log_2 n$

$\Rightarrow T(n) \leq c' \log_2 n$

$\Rightarrow T(n)$ is $O(\log_2 n)$

$\Rightarrow T(n) \geq c\log_2 n$

$\Rightarrow T(n)$ is $\Omega(\log_2 n)$

$\boxed{T(n) \text{ is } \Theta(\log_2 n)}$