Lecture 3

Lecture 3

LAST Time: Binary search, merge sort, quick sort

Divide: Break up the problem into smaller subproblems.

conquer: Solving those subproblem w/same algo.

merge: solutions of subproblems.

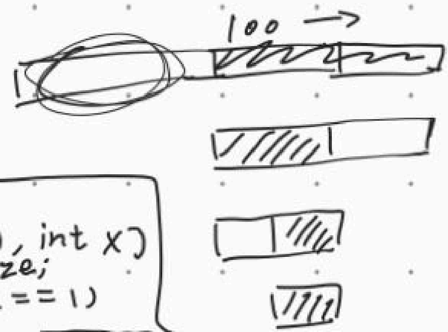
```
int smallest (int A[ ]) is
        smallest A[i]
     [ for (i=2 to n){
O(n) O(1) if (smallest > A[i]) => smallest = A[i]
     [ }
        Return smallest;
}
```

```
int Rsmallest (int A[ ]){
    if (A[i] == 1) return A[i];    O(1)
T(n/2) x = Rsmallest(A[1... n/2]);
T(n/2) y = Rsmallest(A[n/2+1, n];    O(1)
O(1) Return min(x,y);
}
```

$T(n)$ = Return of smallest on input n.

$$T(n) = O(1) + 2T(\tfrac{n}{2})$$

```
bool BS(int A[ ], int x){
        if (A.size==1) Return A[1] == x
        if (x > A[Middle]) => go Right
        else go left
}
```




```
int BS(int A[], int x)
    n = A.size;
    if(A.size == 1)
        Return  A[1] == x
    if(A[n/2] < x)
        BS(A[n/2 ... n]);
    else
        BS(A[1 ... n/2]);
}
```

$$T(n) = O(1) + T\left(\frac{n}{2}\right)$$

$i=0 \quad \frac{n}{2}$

$i=1 \quad \frac{n}{2}$

$i=2 \quad \frac{n}{2^2}$

$i\ ?$

$\boxed{n} \longrightarrow \underbrace{O(1)=1}_{1}$

$\downarrow$

$\boxed{\frac{n}{2}} \longrightarrow \underbrace{O(1)=1}_{1}$

$\downarrow$

$\boxed{\frac{n}{4}} \longrightarrow \underbrace{O(1)=1}_{1}$ $\qquad \sum O(1)$ depth

$\downarrow$

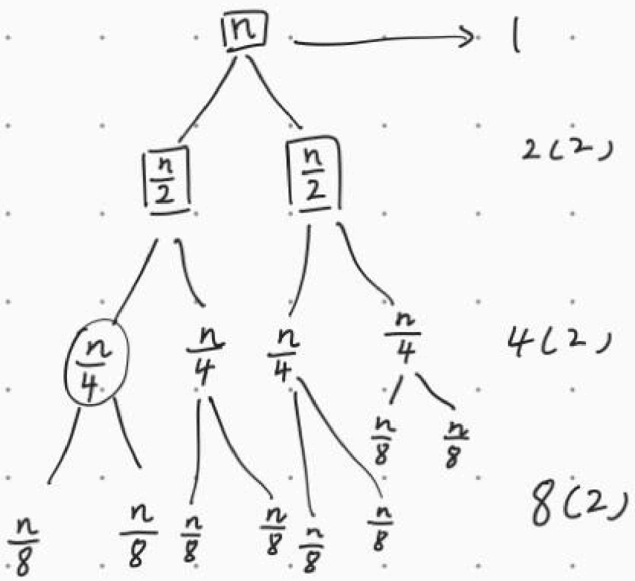
$\boxed{1}$

$$\frac{n}{2^i} = 1 \qquad n = 2^i \qquad \boxed{\log_2 n = i}$$

$$\sum_{i=0}^{\log_2 n} O(1) = O(1)(\log_2 n + 1) = \boxed{O(\log_2 n)}$$

$$\sum_{i=0}^{4} a = 5a$$

$$T(n) = 2T(n/2) + O(1)$$



$\frac{n}{2^i} = 1$

$i = \log_2 n$

$$T(n) = 2T(\frac{n}{2}) + O(1)$$

$$2[2T(\frac{n}{4}) + O(1)] + O(1)$$

$$T(n) = \sum_{i=0}^{\log_2 n} 2^i(2) = 2(2^{\log_2 n + 1}) - 1 = O(n)$$

$$\sum_{i=0}^{x} 2^i = 2^{x+1} - 1 \qquad 2 \cdot 2 \cdot 2^{\log_2 n} - 1$$

$$4n - 1 = O(n)$$

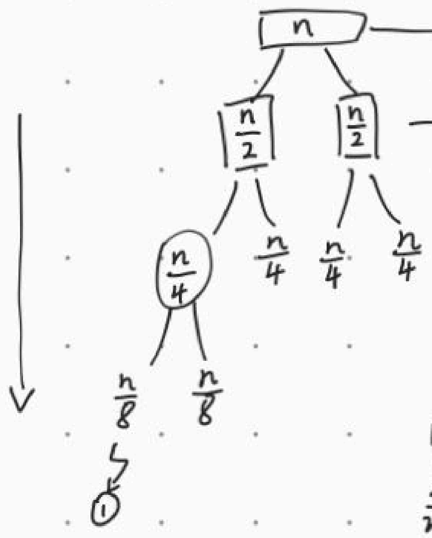
Merge sort

```
int[] ms(int A[]){
  O(1)      n = A.size();
  O(1)      if (n==1) Return A[1];
T(n/2)  int B[] = ms(A[1...n/2]);
T(n/2)  int C[] = ms(A[n/2+1 ...n]);
        Return merge(B,C);  O(n)
}
```

5 items   100 items.   comparison cost · total items

$T(n) = \underline{O(1) + O(1)} + 2T(n/2) + \underline{O(n)}$

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



work

$\rightarrow O(n) = n$

$\rightarrow 2(\frac{n}{2}) = n$

$4(\frac{n}{4}) = n$

work at each level is n

$\sum_{i=0}^{\log_2 n} n = n(\log_2 n + 1)$

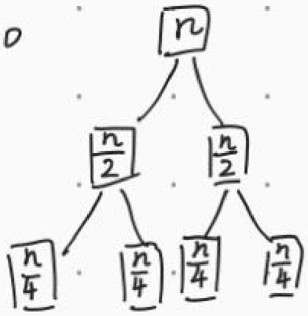
$\frac{n}{2^i} = 1 \quad i = \log_2 n$

$\boxed{\sum_{i=0}^{x} a = a(x+1)}$

```
foo(int A[])
    if(A.size==1) Return
n^2    => foo(i=1 to n^2) print hi;
T(n/2) -> foo(A[1...n/2])
T(n/2) -> foo(A[n/2+1...n])
O(1)   -> Return 1;
}
```

$T(n) = 2T(\frac{n}{2}) + O(n^2)$

$i=0$



work

n^2

$2(\frac{n}{2})^2$

$4(\frac{n}{4})^2$

work: $2^i(\frac{n}{2^i})^2$ depth $i=0$ to $\log_2 n$

$$\sum_{i=0}^{\log_2 n} 2^i(\frac{n}{2^i})^2$$

$$\sum_{i=0}^{\log_2 n} 2^i(\frac{n}{2^i})^2 = \sum_{i=0}^{\log_2 n} 2^i \frac{n^2}{4^i} = \sum_{i=0}^{\log_2 n} n^2(\frac{2}{4})^i$$

$$= n^2 \sum_{i=0}^{\log_2 n} (\frac{1}{2})^i \le n^2 \sum^{\infty}$$

$$\sum_{i=0}^{\infty} (\frac{1}{2})^i \le \frac{1}{1-1/2} = 2$$

$$\sum_{i=0}^{\infty} (\frac{1}{2})^i \le$$

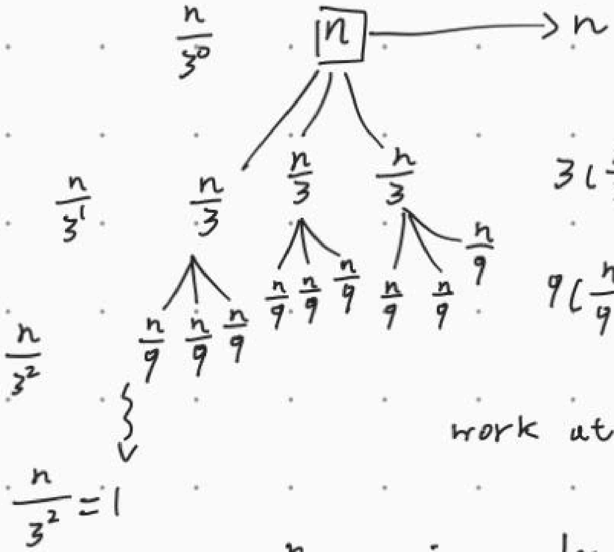
$T(n) = 9T(\frac{n}{3}) + O(n)$ Tree wins

$T(n) = 9T(\frac{n}{3}) + O(n^2)$ Both win

$T(n) = 9T(\frac{n}{3}) + \underline{O(n^3)}$ First work will win

$T(n) = 3T(\frac{n}{3}) + O(n)$

```
foo(int A[])
    for(i=1 to A.size());
            print hello;
    }
foo(A[1....n/3])
foo(A[1... n/3])
foo(A[n/3+1...2n/3])
  Return
}
```



$3(\frac{n}{3}) = n$

$9(\frac{n}{9}) = n$

work at level: n

$\frac{n}{3^i} = 1 \cdot 3^i$

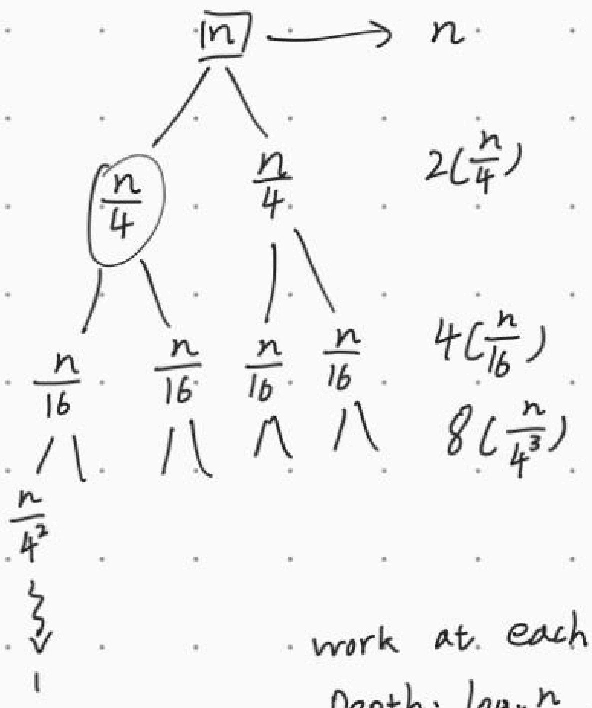
$\log_3 n = \log_3 3^i$

$i = \log_3 n$

$\sum_{i=0}^{\log_3 n} n = n(\log_3 n + 1)$

$= O(n \log n)$

$T(n) = 2T(\frac{n}{4}) + n$



work at each level: $2^i(\frac{n}{4^i})$

Depth: $\log_4 n$

$\frac{n}{4^i} = 1$

$$\sum_{i=0}^{\log_4 n} 2^i \frac{n}{4^i} = n \boxed{\sum_{i=0}^{\log_4 n} (\frac{2}{4})^i} \leq n \sum_{i=0}^{\infty} (\frac{1}{2})^i$$

2

$= 2n = O(n)$

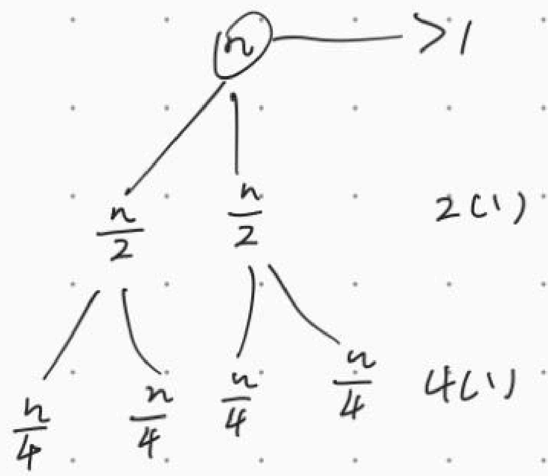
$$\sum_{i=0}^{\infty}\left(\frac{1}{3}\right)^i \le \boxed{\frac{1}{1-r}} = \frac{1}{2/3} = 1.5$$

$$\sum_{i=0}^{x} a^i = \frac{a^{x+1}-1}{a-1} \qquad |a|>1$$

$$\sum_{i=0}^{x} a = a(x+1)$$

$$\sum_{i=0}^{x} r^i \le \sum_{i=0}^{\infty} r^i \le \frac{1}{1-r} \qquad |r|<1$$

$$T(n) = 2T\left(\frac{n}{2}\right) + 1$$



work at level: 2^i

depth: $\log_2 n$

$$\sum_{i=0}^{\log_2 n} 2^i = 2^{\log_2 n + 1} - 1$$

$$= \cancel{\frac{-1}{1-2}}$$

$$2 \qquad = 2n$$