Divide & conquer

1) Divide the bigger problems into smaller subproblems

2) Solve the subproblems (conquer) with the help of Recursion

3) Combine the solutions of all subproblems to get a final solution.

(optional)

Recursion
- Base function — small Problem
- Recursive call

Algorithms

DAC(arr, p, q) { — $T(n)$

Method Name

c — if (small(arr, p, q)) {

return solution(arr, p, q)

}

else {

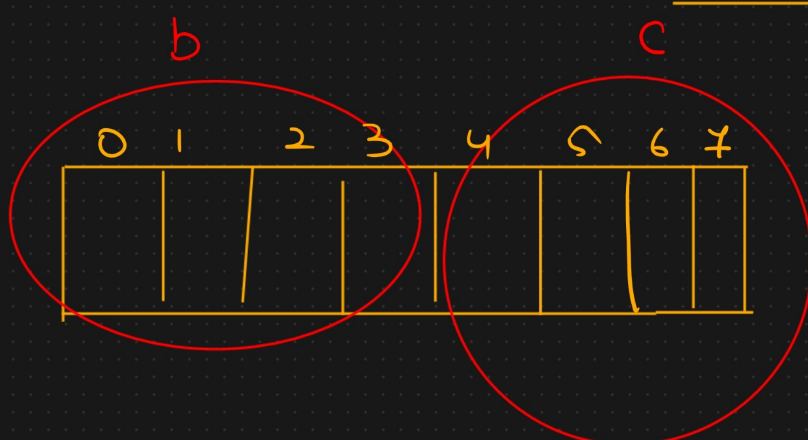
Recursive call

c — m = Divide(arr, p, q)

$T(n/2)$ — b = DAC(arr, p, m);

$T(n/2)$ — c = DAC(arr, m+1, q);

return (combine(b, c)) — c



Recurrence Relation

subproblems

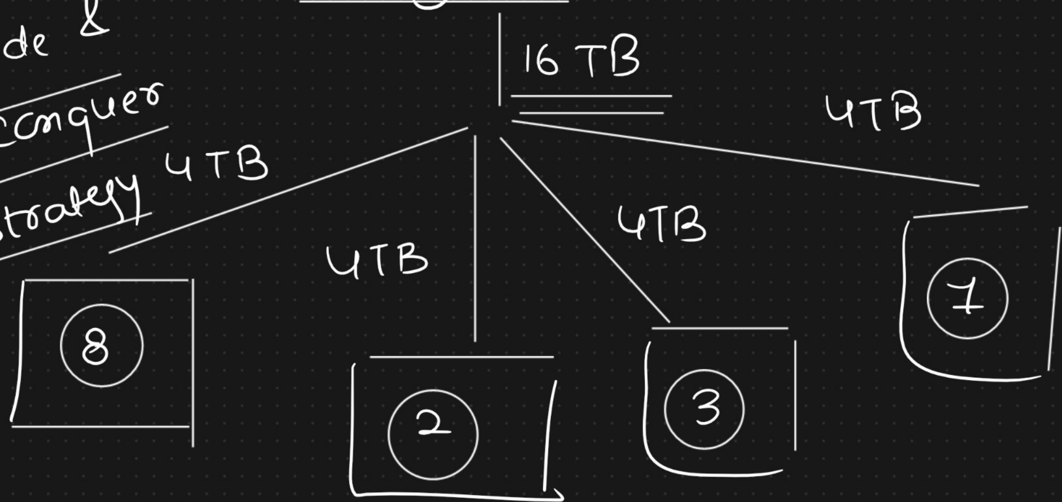
$$T(n) = 2T(n/2) + c$$

size of subproblem

Real time Example

Divide & conquer Strategy

Big Data $(8+2+3+7) = 20$



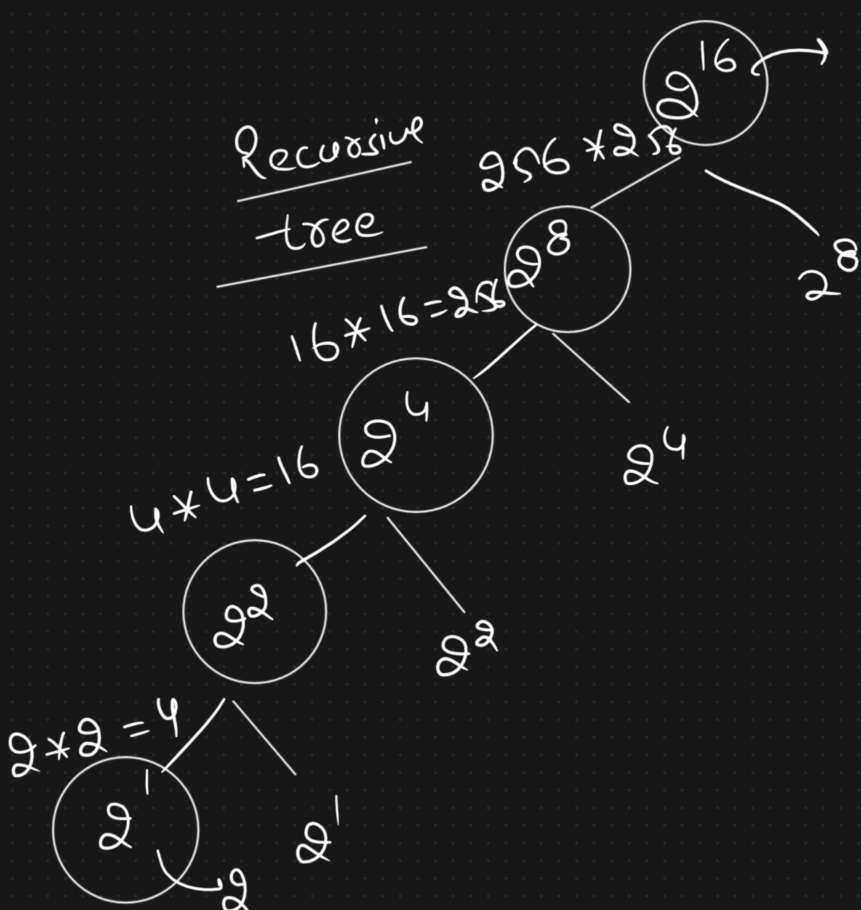
Application of Divide & conquer Strategy

Recursion Module

- Power finding of an element
- Binary Search
  - No combine

$$\begin{cases} a=2 \\ b=16 \end{cases}$$

Recursive tree



```
Power (a, b) {

    if (b == 1) {
        return a;
    }

    else {
        mid = b/2
        c = Power(a, mid)
        result = c * c

        if (b%2 == 0) {
            return result;
        }
        else {
            return result * a;
        }
```

base case

small problem

condition

Bigger Problem

① Divide — mid = b/2

② Conquer — c = Power(a, mid) → $O(\log_2 n)$

③ Combine — result = c * c ... return result * a;

Quicksort

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

Pre-requisites

1) Recurrence Relation Solving
2) Recursion Module