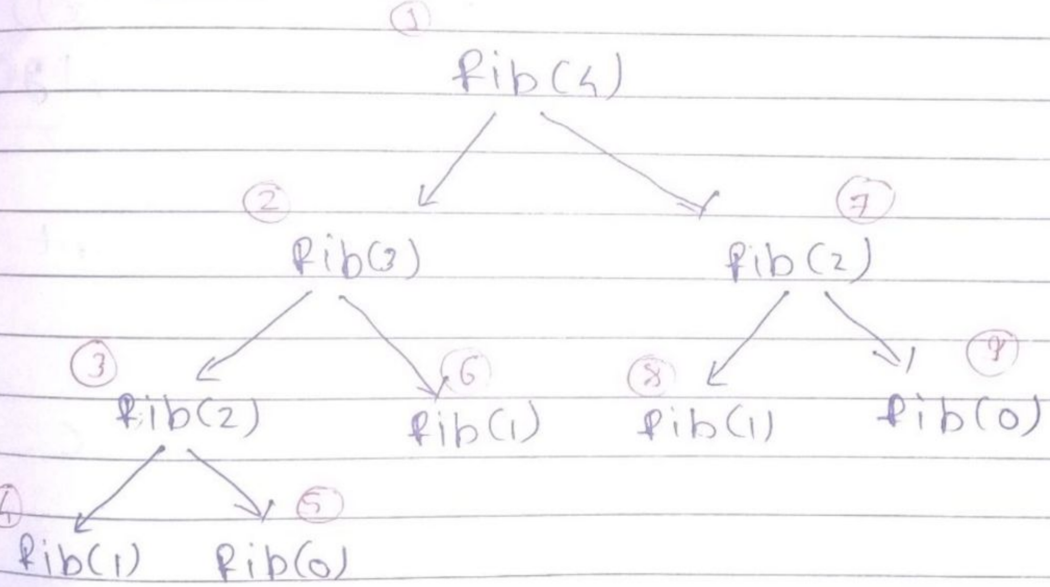
Recursive Algorithms.

In Recursion, we know, ~~fucl~~ function calls are stored in stack. Hence recursive programs don't have constant space complexity.

* space complexity of Fibonacci No —
Let's take fib(4)



Note: • At any particular point of time, no two fuction calls at the same level of recursion will be in the stack at the same time.
• only calls that are interlinked with each other will be in the stack at the same time.

⇒ At one particular level of tree, there will be only one call that are in stack at a time.
It is not possible that all the function calls (Here, 9) will be in the stack at the same time.
maximum space taken = Height of the tree.

Space complexity = O(N)

Page No.
Date :

* Types of Recursions *

1) Linear Recursion 2) Divide and Conquer

* Divide & Conquer Recurrances -

Form :-

$$\boxed{T(x) = a_1 T(b_1 x + \varepsilon_1(x)) + a_2 T(b_2 x + \varepsilon_2(x)) + \cdots + a_k T(b_k x + \varepsilon_k(x)) + g(x)}$$

for $x \geq x_0$ ← constant

For Ex. In case of binary search.

$T(N) = T(N/2) + C$

comparing with above eq^n

Here, $a_1 = 1$, $b_1 = \frac{1}{2}$, $\varepsilon_1(n) = 0$, $g(x) = C$

* Best way to get complexity *

* Akra - Bazzi Formula *

$$\boxed{T(x) = \Theta\left(x^p + x^p \int_1^x \frac{g(u)}{u^{p+1}}\, du\right)}$$

Here, $T(x)$ - Time complexity of x.

What is P ?

$a_1 b_1^p + a_2 b_2^p + \cdots = 1$

or

$$\boxed{\sum_{i=1}^{k} a_i b_i^p = 1}$$

Page No.
Date :

Example — $T(N) = 2T\left(\frac{N}{2}\right) + (N-1)$

Here, $a_1 = 2$, $b_1 = 1/2$, $g(N) = N-1$

put in the formula of P & find P

$$2 \times \left(\frac{1}{2}\right)^P = 1$$

$$\boxed{P = 1}$$

Now, put P in Akra Bazzi formula

$$T(x) = \Theta\left[x^1 + x^1 \int_1^x \frac{u-1}{u^2}\, du\right]$$

$$= \Theta\left[x + x \int \left(\frac{1}{u} - \frac{1}{u^2}\right) du\right]$$

$$= \Theta\left[x + x\left[\log u + \frac{1}{u}\right]_1^x\right]$$

$$= \Theta\left[x + x\left(\log x + \frac{1}{x} - 1\right)\right]$$

$$= \Theta\left[x + x\log x + 1 - x\right]$$

$$= \Theta[x\log x + 1) \quad \text{// Ignore constants}$$

$$= \Theta(x \log x)$$

Here for array of size N:

Merge sort complexity $= \Theta(N\log N)$

If you can't find value of P :-

$$T(x) = 3T\left(\frac{x}{3}\right) + 4T\left(\frac{x}{4}\right) + x^2$$

Let's try P = 1

$$3 \times \left(\frac{1}{3}\right)^1 + 4 \times \left(\frac{1}{4}\right)^1 = 1$$

$2 > 1 \Rightarrow$ Increase denominator

Page No.
Date :

$2 > 1$ → Now we have to increase the denominator i.e. increase the the value of P

$\boxed{P > 1}$

Let's try $P = 2$

$$3 \times \frac{1}{9} + 4 \times \frac{1}{16}$$

$$= \frac{1}{3} + \frac{1}{4} = \frac{4+3}{12} = \frac{7}{12} < 1$$

$\therefore \boxed{P < 2}$

Note :- when, P < Power of $g(x)$
then, ans = $g(x)$

Here,

$g(x) = x^2$

& $P < 2$ (i.e. Power of $g(x)$)

Hence, ans = $O(g(x)) = O(x^2)$

** Linear Recurrances **

Form of Homogeneous Linear Recurrances.

$$f(x) = a_1 f(x-1) + a_2 f(x-2) + a_3 f(x-3) + \text{---} \text{---} + a_n f(x-n)$$

or

$$f(x) = \sum_{i=1}^{n} a_i f(x-1), \text{ for } a_i \text{ \& } n = \text{fixed}$$

n = order of recurrance.

Page No.
Date :

Fibonacci Example:-

$$f(n) = f(n-1) + f(n-2) \qquad \text{———} \quad (1)$$

sol^n:-

Step 1:- Put $f(n) = \alpha^n$ for some const. in eq^n (1)

$\Rightarrow \alpha^n = \alpha^{n-1} + \alpha^{n-2}$.

$\Rightarrow \boxed{\alpha^n - \alpha^{n-1} - \alpha^{n-2} = 0.}$ - charateristic eq^n of recurrance.

divide by α^{n-2}.

$$\alpha^2 - \alpha - 1 = 0.$$

roots are :- $\alpha = \dfrac{1 \pm \sqrt{5}}{2} \quad \left(\dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}\right)$

$\alpha_1 = \dfrac{1+\sqrt{5}}{2}, \quad \alpha_2 = \dfrac{1-\sqrt{5}}{2}$

Step 2:-

$f(n) = c_1 \alpha_1^n + c_2 \alpha_2^n + \text{----} + c_n \alpha_n^n$.

Here,

$$f(n) = c_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + c_2 \left(\frac{1-\sqrt{5}}{2}\right)^n \qquad \text{———} \quad (2)$$

Step 3:- Fact.

* no of roots = no. of answers we have already

Here, we have 2 roots α_1 & α_2.
Hence we should have 2 answers already.
$\therefore F(0) = 0$ & $f(1) = 1$.

Page No.
Date :

putting in eq^n(2)

$F(0) = 0$

$\Rightarrow c_1 + c_2 = 0 \Rightarrow \boxed{c_1 = -c_2}$ ——— (3)

$F(1) = 1$

$\Rightarrow c_1\left(\frac{1+\sqrt{5}}{2}\right) + c_2\left(\frac{1-\sqrt{5}}{2}\right) = 1$

from (3)

$c_1\left(\frac{1+\sqrt{5}}{2}\right) - c_1\left(\frac{1-\sqrt{5}}{2}\right) = 1$

$\therefore \boxed{c_1 = \frac{1}{\sqrt{5}}}, \boxed{c_2 = -\frac{1}{\sqrt{5}}}$

putting in eq^n(2)

$$\boxed{f(n) = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n}$$

$$f(n) = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$$

as $n \to \infty$
- this will be close to 0
Hence this is less dominating term
Hence ~~ing ing~~ ignore while taking complexity

$\therefore$ Time complexity for ~~nn~~ n^{th} Fibonacci Numbers.

$= O\left(\frac{1+\sqrt{5}}{2}\right)^n$ → This is also called 'Golden Ratio'

Page No.
Date :

* When we get Equal roots ——

→ If α is repeated r times then $\alpha^n, n\alpha^n, n^2\alpha^n, \ldots\ldots, n^{r-1}\alpha^n$ are all solution to the recurrance

* Non-homogeneous linear Recurrances *

* Form –

$$f(n) = a_1 f(n-1) + a_2 f(n-2) + a_3 f(n-3) + \ldots$$
$$\ldots + a_d f(n-d) + g(n)$$

↳ when this extra funct^n is there it is non-homogeneous

Sol^n —

Replace $g(n)$ by 0 & solve usually

Example – $f(n) = 4f(n-1) + 3^n$ —— (i)

Step 1 – put $g(n) = 0 \Rightarrow 3^n = 0$.

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

put $f(n) = \alpha^n$.

$$\alpha^n = 4\alpha^{n-1}$$
$$\alpha^n - 4\alpha^{n-1} = 0$$

Divide by $n-1$

$$\alpha - 4 = 0$$
$$\therefore \boxed{\alpha = 4}$$

Step 2 –

Homogeneous sol^n :- $f(n) = c_1\alpha^n$ // put $\alpha = 4$

$$\boxed{f(n) = c_1 4^n}$$

Page No.
Date :

Step3 :- Take $g(n)$ on one side & find particular sol^n —

$\therefore f(n) - 4f(n-1) = 3^n$ —— (2)

** Guess something that is similar to $g(n)$

My Guess : $f(n) = C3^n$ —— (3)

put in eq^n (2) —

$C3^n - 4C3^{n-1} = 3^n$

Divide by $n-1$

$\boxed{C = -3}$

Find particular sol^n

put $C = -3$ in eq^n (3) —

$\Rightarrow f(n) = -3 \times 3^n = -3^{n+1}$

Step4 :- Find general sol^n by adding both the sol^n

$F(n) = C_1 4^n + (-3^{n+1})$ —— (4)

we already have two answers

i.e. $f(0) = 0$ & $f(1) = 1$

$\Rightarrow f(1) = 1$

$C_1 4 - 3^2 = 1$

$\boxed{C_1 = 5/2}$

Put the value of C_1 in eq^n (4)

$$\boxed{F(n) = \frac{5}{2}4^n - 3^{n+1}}$$

$\therefore$ Time complexity $= O\left(\frac{5}{2}4^n - 3^{n+1}\right)$

* How do we guess a particular solⁿ?

* If $g(n)$ is exponential :
  Then guess of same type.
  Ex :- $g(n) = 2^n + 3^n$
  Guess :- $\boxed{f(n) = a2^n + b3^n}$

* If $g(n)$ is polynomial :
  then guess of same degree.
  Ex :- $g(n) = n^2 - 1$
  Guess of same degree (Here degree = 2)
  Guess :- $f(n) = an^2 + bn + c$.

* If $g(n)$ is combination of exponential & polynomial
  Ex :- $g(n) = 2^n + n$
  Guess :- $f(n) = a2^n + (bn + c)$

Note :- let say you guessed, $F(n) = a2^n$
and it fails then try $(an+b)2^n$.
If this also fails, increase the degree.
i.e. $(a^2n + bn + c)2^n$