05-04

Monday, May 4, 2020 6:18 PM

Fib. number

0, 1, 1, 2, 3, 5, 8, 13, 21, ...

a_0 a_1 a_2 a_3 a_4 ......

$\boxed{a_n = a_{n-1} + a_{n-2}}$ Q: design algo to compute a_n

fib(int n) fib(1) ⇒ 1

fib(4) ⇒ 3

```
fib1 (int n) {
    if (n==0 || n==1) {
        return n;
    }
    return fib(n-1) + fib(n-2);
}
```

T1 T2

corectness efficient??

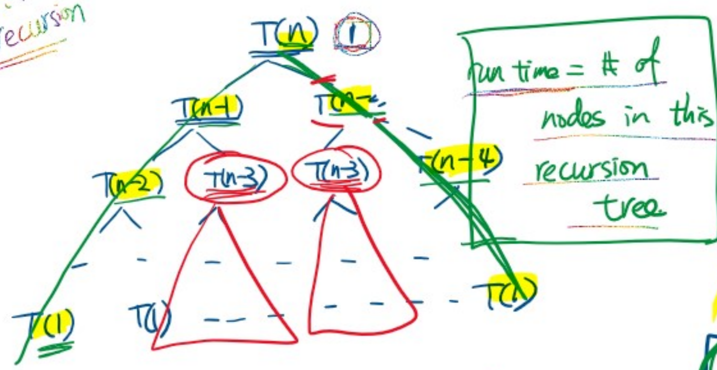
running time space

big-O exponatial

$T(n)$:= running to compute fib1(n)

$T(n) = T(n-1) + T(n-2) + 1$ constant

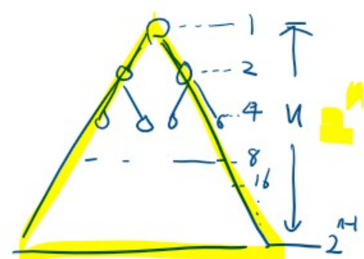
T(n) = ?? solve recursion



$$\boxed{2^{\frac{n}{2}} \le T(n) \le 2^n}$$

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

1.4^n

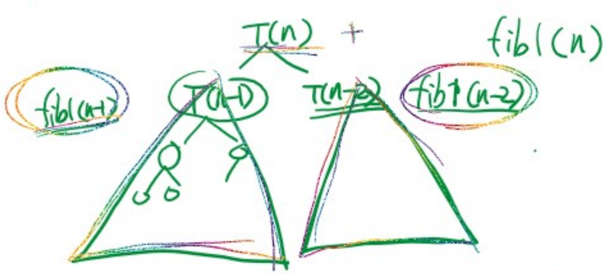


$1+2+2^2+2^3+2^4+\cdots+2^{n-1}$

geometric

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

$\sim 2^n$



why is slow? same computation over and over again why is slow? same computation over and over again

a_{n-3} a_{n-3}

```
fib2(n) {
    A[0] = 0;
    A[1] = 1;
    for (i=2 ..... n) {
        A[i%2] = A[(i-1)%2] + A[(i-2)%2]
    }
    return A[n%2]
}
```

fib2(0)
fib2(1)

A [0 | 1]

$A[i] = fib2(i)$

r.t. $O(n)$ space. $O(n)$

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} a_2 & a_1 \\ a_1 & a_0 \end{bmatrix}$$

$$A^2 = A \times A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} a_3 & a_2 \\ a_2 & a_1 \end{bmatrix}$$

$$A^3 = A^2 \times A = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} a_4 & a_3 \\ a_3 & a_2 \end{bmatrix}$$

$$A^k = \begin{bmatrix} a_{k+1} & a_k \\ a_k & a_{k-1} \end{bmatrix}$$

a_k

fib3(n) $\Longrightarrow$ $\boxed{A^n}$

$f(n) = x^n$

$\boxed{x \cdot x \cdot x \cdot \ldots \cdot x}$ $\underbrace{}_{n-1}$ $O(n)$

$O(\lg n)$

$$\boxed{f(n) = \begin{cases} \left[f\left(\frac{n}{2}\right)\right]^2 & n \text{ is even} \\ \left[f\left(\frac{n-1}{2}\right)\right]^2 \cdot x & n \text{ is odd} \end{cases}}$$

r.t. $O(\lg n)$

space $O(\lg n)$

$f(64) = x^{64}$ #1 63 multiplication

$\hookrightarrow [f(32)]^2 \to [[f(16)]^2]^2 \to [[[f(8)]^2]^2]^2$

$(((f(2)^2)^2)^2)^2 \leftarrow ((f(4)^2)^2)^2$

$((((((x)^2)^2)^2)^2)^2)^2$ #2 6 multiplications

$(((((x^2)^2)^2)^2)^2)^2$ #2 6 multiplications

fib1 $\Rightarrow$ fib2 $\Rightarrow$ fib3

	fib1	fib2	fib3	
	$O(2^n)$	$O(n)$	$O(\log n)$	r.t.
	$O(n)$	$O(1)$	$O(\log n)$	space

break to 8:15

asyn. notations

$\boxed{\text{cmp growth of function.}}$ $\boxed{\text{cmp numbers}}$

$f(n)$ $g(n)$	a b
$f(n) = O(g(n))$	$\leq$
$f(n) = o(g(n))$	$<$
Θ	$=$
Ω	$\geq$
ω	$>$

$f(n) = 2n^2$ $g(n) = n^3 - 5$

$f(n) = O(g(n))$

$f(n) = o(g(n))$

$g(n) = \Omega(f(n))$

$g(n) = \omega(f(n))$

logarithm: $\log n$, $(\log n)^2$

polynormial: n n^2 n^3, $\sqrt{n} = n^{\frac{1}{2}} = n^{0.5}$

exp.: 14^n 2^n 3^n

$f(n) = n \lg n$ $g(n) = (\log n)^2$

2.78 3 n

$f(n) = n(\lg n)$ $\quad g(n) = (\lg n)$

$\log n \quad n\log n \quad \overbrace{n \quad n^2 \quad n^{2.78} \quad n^3} \quad 2^n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} \rightarrow \begin{cases} \infty & f(n) \text{ is much faster} \\ & f(n)=\Omega(g(n)) \quad f(n)=\omega(g(n)) \\ C & f(n) \text{ is about same as } g(n) \\ & f(n)=O(g(n)) \quad f(n)=\Omega(g(n)) \quad f(n)=\Theta(g(n)) \\ 0 & g(n) \text{ is much faster} \\ & f(n)=O(g(n)) \quad f(n)=o(g(n)) \end{cases}$$

$f(n) = (\log n)^2$

logarithm

$f(n) = \log n \qquad g(n) = (\log n)^2$

$f(n) = O(g(n))$

$f(n) = o(g(n))$

$f(n) = \log_2 n. \quad g(n) = \log_3 n$

$f(n) = \Theta(g(n))$

$$\log_a b = \frac{\log_c b}{\log_c a}$$

$$\lim_{n\to\infty} \frac{\log_2 n}{\log_3 n} = \log_3 2$$

$T(n) = O(\lg n)$

$f(n) = 2^n \qquad g(n) = 3^n$

$f(n) = O(g(n))$, o

$$\lim_{n\to\infty} \frac{2^n}{3^n} = \lim_{n\to\infty} \left(\frac{2}{3}\right)^n \rightarrow 0$$

$T(n) = O(f(n)) \qquad T(n) = \Theta(f(n))$

---

solve recursion

- recursion tree [general, take time]
- ✓ $\boxed{\text{master's thm}}$ [limited, fast]

$$\boxed{T(n) = aT\left(\frac{n}{b}\right) + n^c} \Leftarrow$$

$$T(n) = aT(\tfrac{n}{b}) + n^c$$

merge sort ⟹ e.g. $T(n) = 2T(\frac{n}{2}) + n$

Strassen ⟹ $T(n) = 7 \cdot T(\frac{n}{2}) + n^2$

$a=7 \quad b=2 \quad c=2$

$\log_2 7$ vs 2

$T(n) = O(n^{\log_2 7})$

$\log_b a$ vs c

① $\log_b a > c$ $\qquad T(n) = O(n^{\log_b a})$

② $\log_b a = c$ $\qquad T(n) = O(n^c \cdot \log n)$

③ $\log_b a < c$ $\qquad T(n) = O(n^c)$

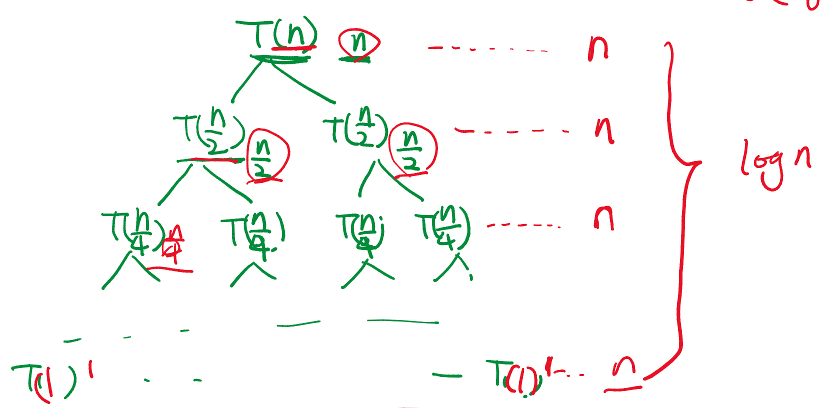
$T(n) = T(\frac{n}{2}) + 1$ $\qquad a=1 \quad b=2 \quad c=0$

$\log_b a = \log_2 1 = 0$

$T(n) = O(\log n)$

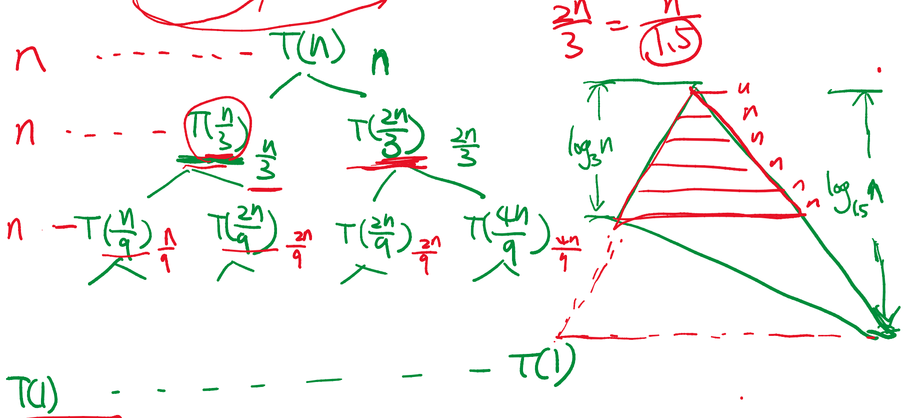
$T(n) = 2T(\frac{n}{2}) + n$

$T(n) = O(n \lg n)$



$T(n) = T(\frac{n}{3}) + T(\frac{2n}{3}) + n$ $\Leftarrow$ quick sort

$\frac{2n}{3} = \frac{n}{1.5}$



$\boxed{\log_2 n}$

$$\underbrace{n \log_3 n}_{n \log n} \le T(n) \le \underbrace{n \cdot \log_{1.5} n}_{n \log n}$$

$n \log n$ $n \log n$

$\boxed{T(n) = T(n-1) + n}$ $\boxed{T(n) = O(n \lg n)}$