CSL 356 July 24

Computing Fibonacci Nos.

i.e. given $\underline{n}$, compute F_n

where $F_0 = 0$ $F_1 = 1$ $F_i = F_{i-1} + F_{i-2}$ otherwise

Method I : write the equivalent recursive program

Method II $F_2 = 0 + 1 = 1$ $F_3 = F_2 + F_1$

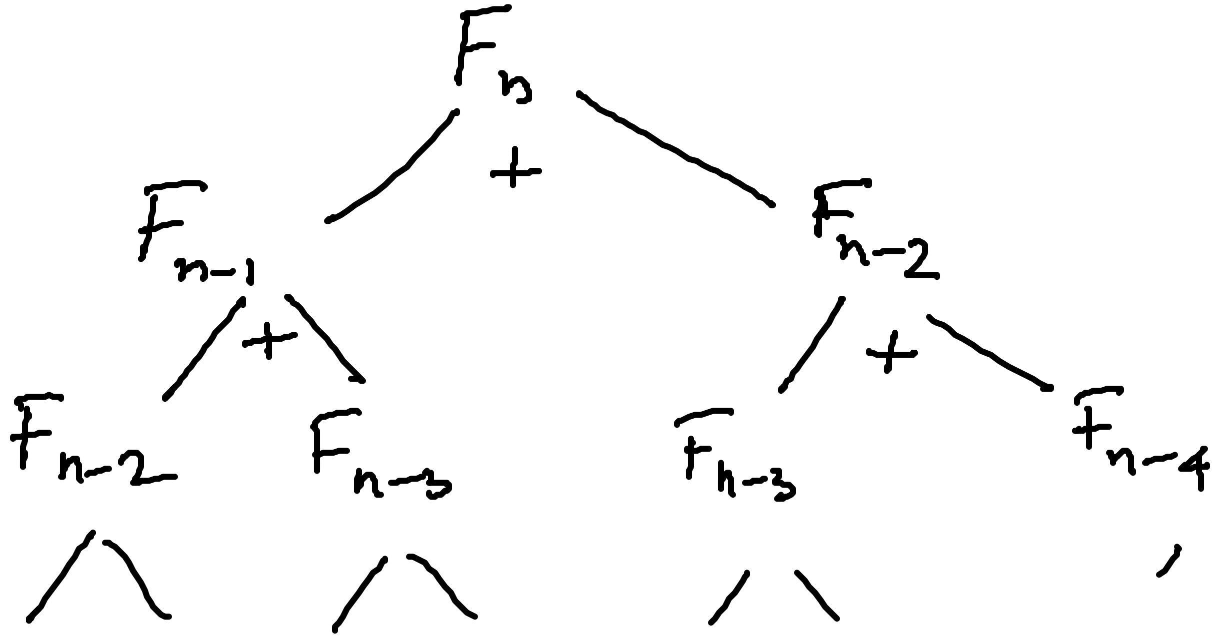
.... F_n :

Time for Method 2

$T^{II}(n)$: the no. of steps (# instructions executed) for computing F_n using Meth II

About n iterations where in each iteration we sum two previously computed Fib nos.

$\Rightarrow$ $O(n)$ additions Space : 2 nos.



$F_1 \quad F_0 \quad \cdots \quad F_0 \quad \cdots$

$$T^I(n) = T^I(n-1) + T^I(n-2) + 1$$

to compute F_{n-1} recursively

F_{n-2} rec.

addition

$T(1) = 1 \qquad T(0) = 1$

$T^I(n) = ?$ atleast F_n which is roughly $(1.6)^n$ additions

~~Method~~ I

Cost of addition?

In Method II, if you consider the last $\frac{n}{2}$ iterations, we are adding nor. of size $\frac{n}{2}$ bits

Adding two b bit nos takes $O(b)$ steps

The last $\frac{n}{2}$ iterations cost $\Theta(n)$ steps

$\Rightarrow \Omega(n^2)$ steps

actually $O(n^2)$ steps overall

What is the min time to compute F_n?

Any algorithm must take time $\propto$ input size + output size

$\Omega(n)$ is a lower bound for F_n

$$\begin{bmatrix} F_i \\ F_{i-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_{i-1} \\ F_{i-2} \end{bmatrix}$$

2x2

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_{i-2} \\ F_{i-3} \end{bmatrix}$$

$$A \cdot A \cdot A \cdots \begin{bmatrix} F_1 \\ F_0 \end{bmatrix}$$

$$F_n = A^{n-1} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

What is the time to compute x^n

$$x^n = \left(x^{\frac{n}{2}}\right)^2 \begin{cases} \left(x^{\frac{n}{2}}\right)^2 & \text{if } n \text{ is even} \\ x \cdot \left(x^{\frac{n-1}{2}}\right)^2 & \text{otherwise} \end{cases}$$

$\log n$ multiplications : but what size?

$$|x^n| = n \log_2 x$$