

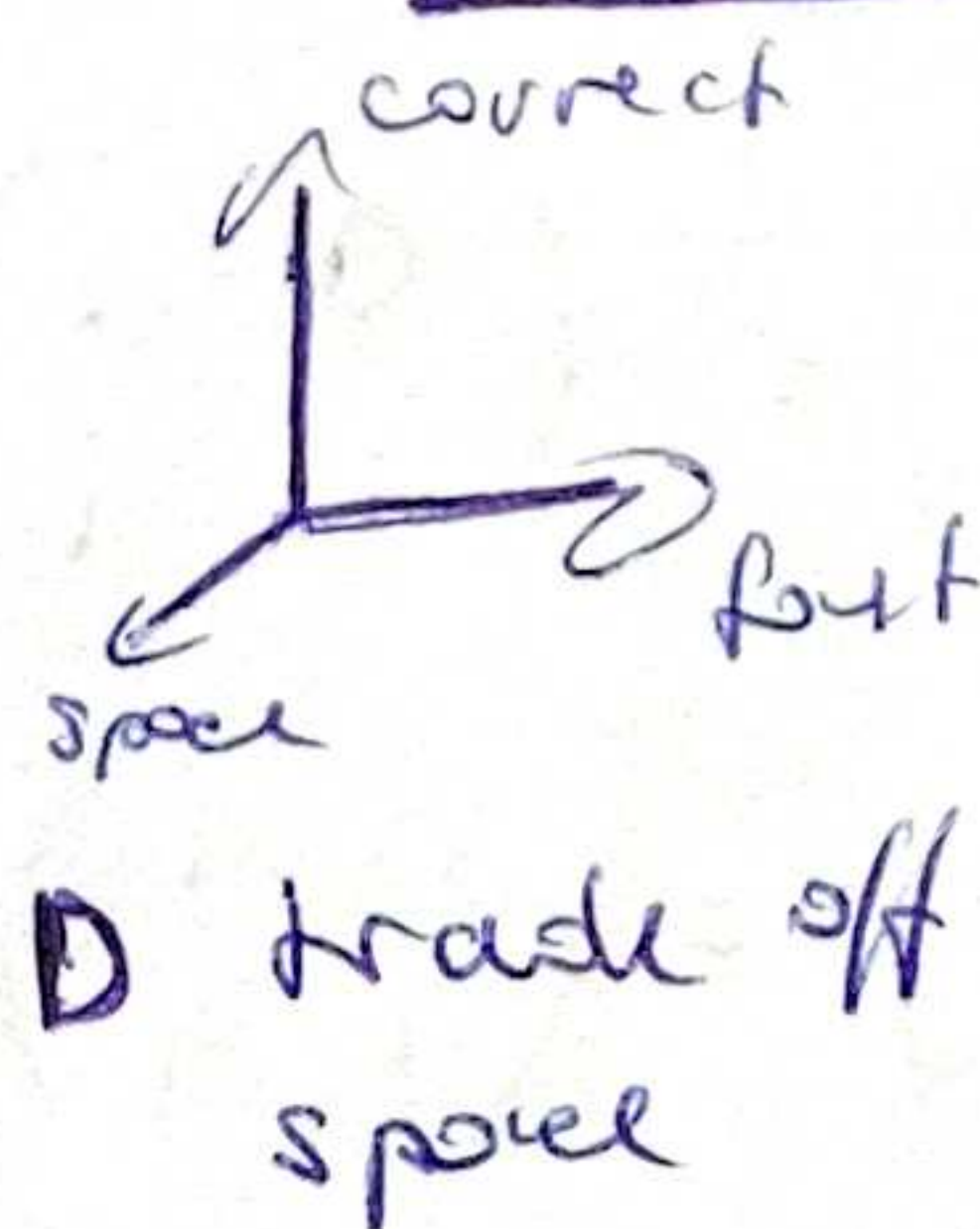
Multiply Alg.

①

$$\begin{array}{r}
 74 \quad 19 \\
 148 \quad 9 \\
 \hline
 296 \quad 4 \\
 592 \quad 2 \\
 \hline
 1184 \quad 1 \\
 \hline
 1406
 \end{array}$$

Algorithms

- ① Does it stop?
- ② Is it correct?
- ③ How fast is it?
- ④ How much space it takes?



②

74
100 11 (19)

multiply with
the sum of powers of 2

Problem: Calculating Fibonacci #s

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

at least doubling every

2 steps → exponential growth

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0, F_1 = 1$$

$$F_n \geq 2^{n/2}, n \geq 6$$

Exponential

Fib(n)

$0 \leq n$

if $n=0$ return 0

else if $n=1$ return 1

else return $F(n-1) + F(n-2)$

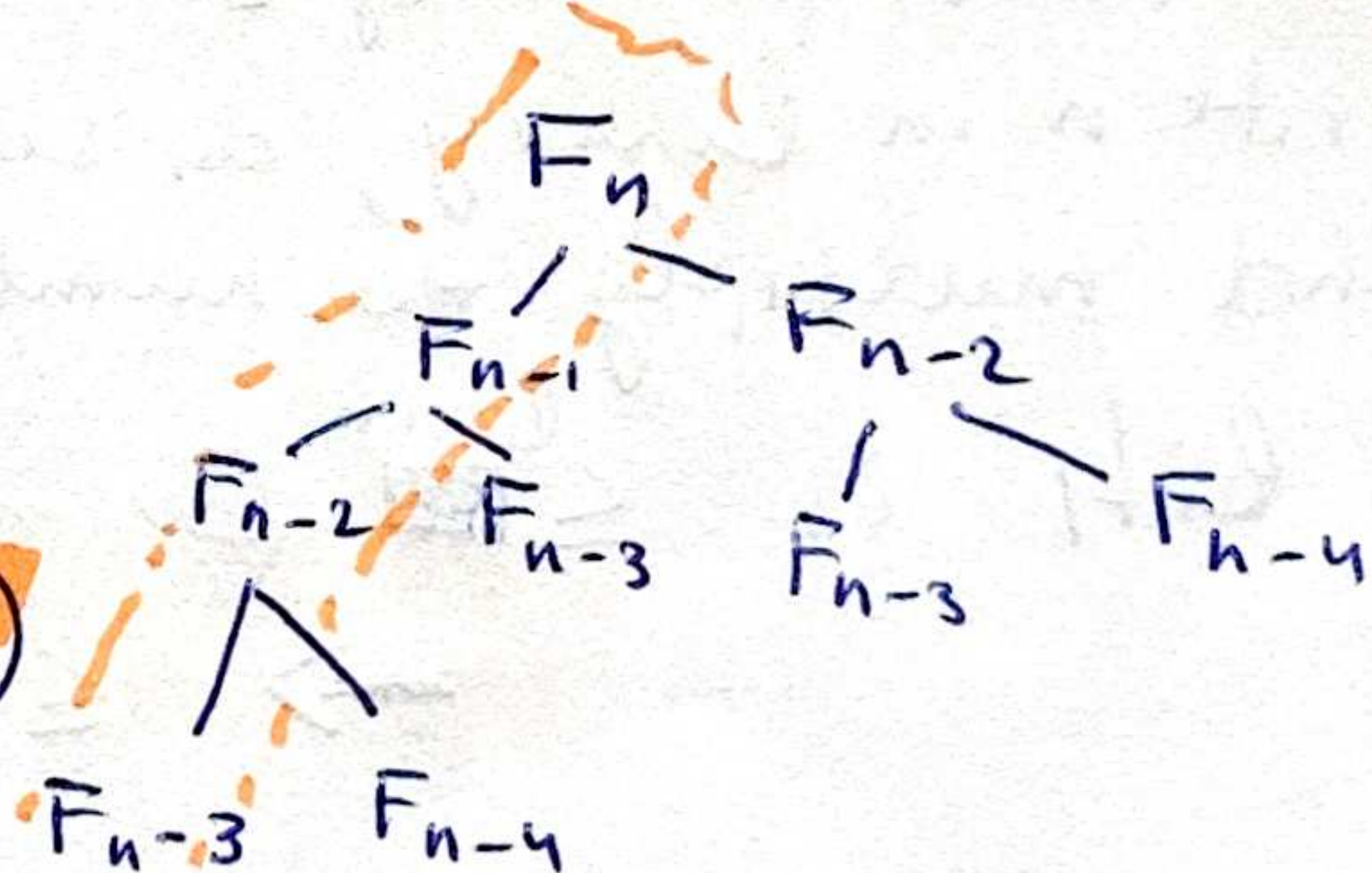
Time

let $T(n) = \#$ additions

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

$$T(0) = 0, T(1) = 0$$

$$T(2) = 1$$



- potentially exponential

space: - depends on recursion implementation

- can put on the stack etc...

Iterative

$0 \leq n$

Fib2(n)

$$F[0] = 0$$

$$F[1] = 1$$

for $i=2$ to n do

$$F[i] = F[i-1] + F[i-2]$$

return $F[n]$

Time

$n-1$ additions

space

- linear

- or constant, 3 cell array

initially small
then big #s.

→ $O(n^2)$ considering growth in bits

3 matrix reduce fibonacci problem into matrix powering prob

$$\begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

$$\begin{pmatrix} F_2 \\ F_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^2 \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

Consider raising a number to a power.

$3^n \rightarrow$ n multiplications
does not help, already have $n-1$ additions

case let n be a power of 2

$$n = 2^k$$

$\log n = k$ multiplications

$$3 = 3$$

$$3^2 = 9$$

$$3^4 = 81$$

$$3^8 = 7209$$

} repeated squaring

case n is not power of 2

write n in binary, as sum of powers of 2
and multiply the numbers taken to these powers

of bits is $\leq \log n$ for a number n

$\Rightarrow \leq \log n$ multiplications in the final step

$\leq \log n$ multiplications for each primary step

$\rightarrow \leq 2 \log n$ in total

Consider matrices

each matrix is a fixed # of mult and additions

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} F_0 \\ F_1 \end{pmatrix} \quad \begin{matrix} 4 \text{ mult} \\ 2 \text{ add} \end{matrix}$$

$$\rightarrow \leq \underbrace{(4 \text{ mult} + 2 \text{ add})}_{C} 2 \log n$$

But representing n th fibonacci # takes

$$F_n \geq 2^{n/2} \text{ for } n \geq 6$$

$$\text{bits}(F_n) \geq \frac{n}{2} \text{ addition of } 2$$

then not faster iter.

$$F_n^2$$

$$C \cdot (\log n) \cdot (n + M(n))$$

time to multiply

Formula

still repeated squaring

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

↑
real numbers

whereas matrix is integer

↳ latter may have a better implementation

Inductionif a statement $P(n)$ holds

← base case

for $n=1$, and if for every $n \geq 1$ $P(n) \Rightarrow P(n+1)$

← IH

then $P(n)$ holds $\forall n \geq 1$ ex 1

$$S(n) = \sum_{i=1}^n i$$

Prove

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

Base case: true for $n=1$

$$1 = \frac{1 \cdot 2}{2}$$

IH: assume true for n

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

reduction: show true for $n+1$

to IH

$$S(n+1) = \underset{\substack{\uparrow \\ \text{by def.}}}{S(n)} + (n+1) = \underset{\substack{\uparrow \\ \text{by IH}}}{\frac{n(n+1)}{2}} + (n+1) \text{ by IH}$$

$$= \frac{(n+1)(n+2)}{2} \quad \checkmark$$

philosophy of O notation

- not care about constant factors
- care about asymptotics as the problem size gets big

Defn.

We say for ~~positive (integer) function~~ positive (integer) functions

$f(n)$ and $g(n)$ that $f(n)$ is $O(g(n))$

if \exists constants c, N

s.t. $\forall n \geq N$

$$f(n) \leq cg(n)$$

ex $2n^3 + 4n^2$ is $O(n^3)$ ✓

for $n \geq 1 \leftarrow N$

$$2n^3 + 4n^2 \leq 6n^3$$

ex $2^{0.001n}$ is $O(n^{1000})$

NO

ex $10 \log_2 n$ is $O(\ln n)$ ✓

$\log_2 e \cdot \log_2 n = \log_2 n$

$$\lim_{n \rightarrow \infty} \frac{n^{1000}}{2^{0.001n}} = 0$$

ex $n^{1/1000}$ is $O((\log_2 n)^{1000})$ NO

$$m = \log_2 n$$

$$2^{0.0001m} \text{ is } O(m^{1000})$$