CS473: Algorithms

https://courses.engr.illinois.edu/cs473

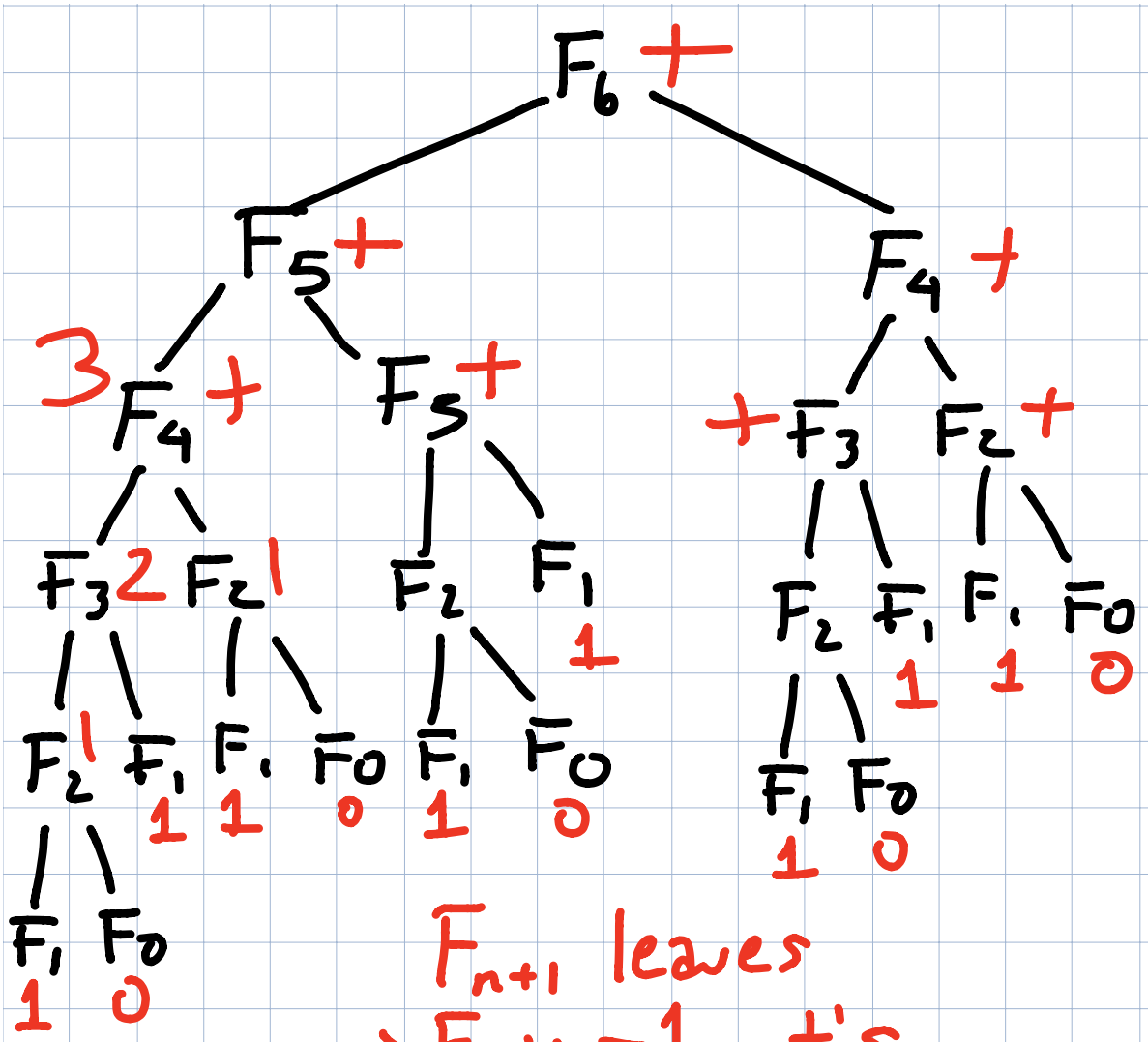
Fibonacci #s:

$F_0 = 0$

$F_1 = 1$

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

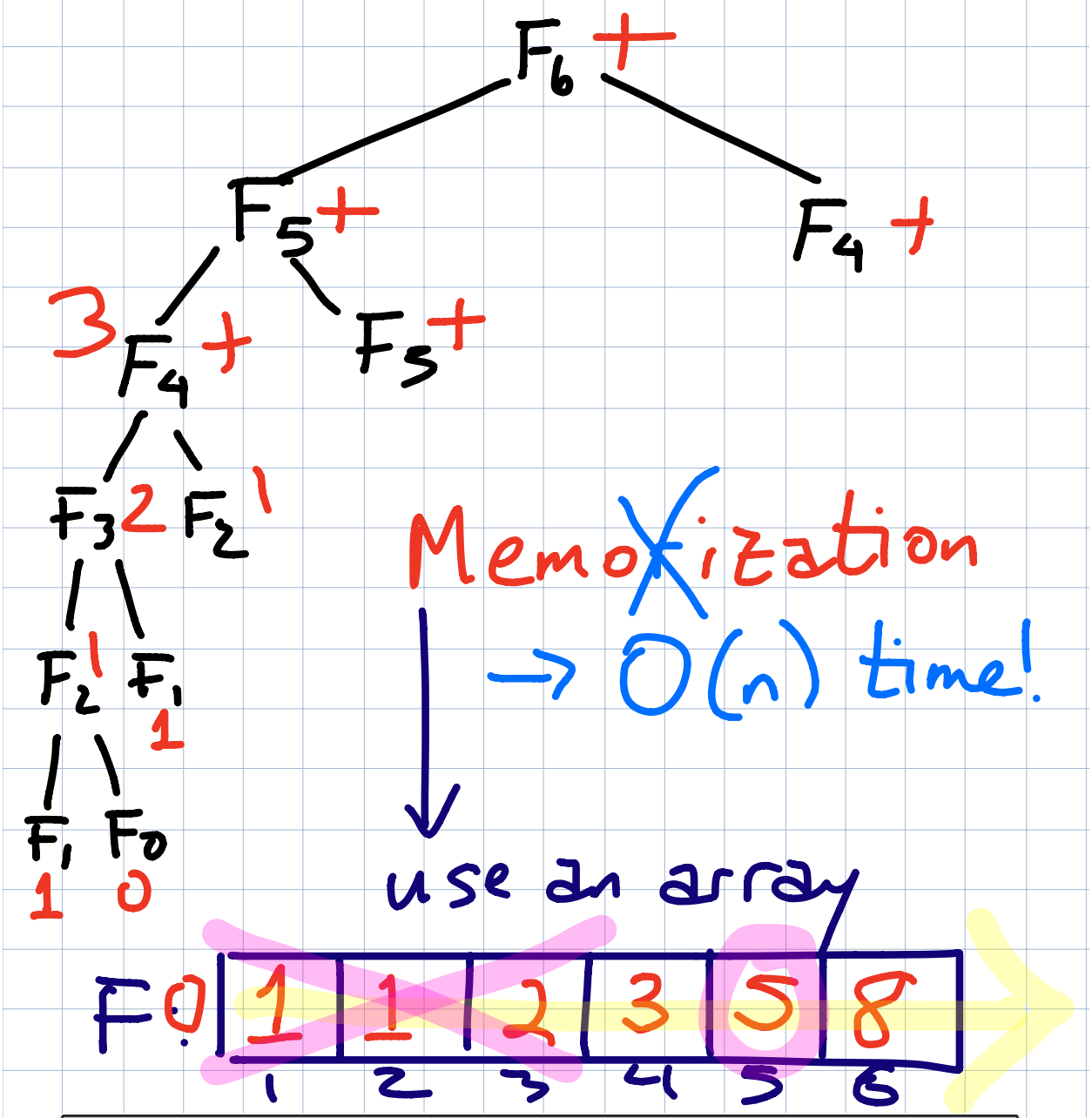
```
RecFibo(n):
  if (n < 2)
      return n
  else
      return RecFibo(n − 1) + RecFibo(n − 2)
```



F_{n+1} leaves

→ $F_{n+1} - 1$ +'s

→ $O(F_n)$ time




```
MemFibo(n):
  if (n < 2)
      return n
  else
      if F[n] is undefined
          F[n] ← MemFibo(n − 1) + MemFibo(n − 2)
      return F[n]
```

```
ITERFIBO(n):
  F[0] ← 0
  F[1] ← 1
  for i ← 2 to n
      F[i] ← F[i−1] + F[i−2]
  return F[n]
```

Dynamic Programming

1. Recurrence
2. Memoize
   - Subproblems
   - Data Structure
3. Order, Purpose
   - Dependencies
   - Eval. Order
4. Time, Space

5. Optimize!!

```
ITERFIBO2(n):
    prev ← 1
    curr ← 0
    for i ← 1 to n
        next ← curr + prev
        prev ← curr
        curr ← next
    return curr
```

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} prev \\ curr \end{bmatrix} = \begin{bmatrix} curr \\ next \end{bmatrix}$$

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

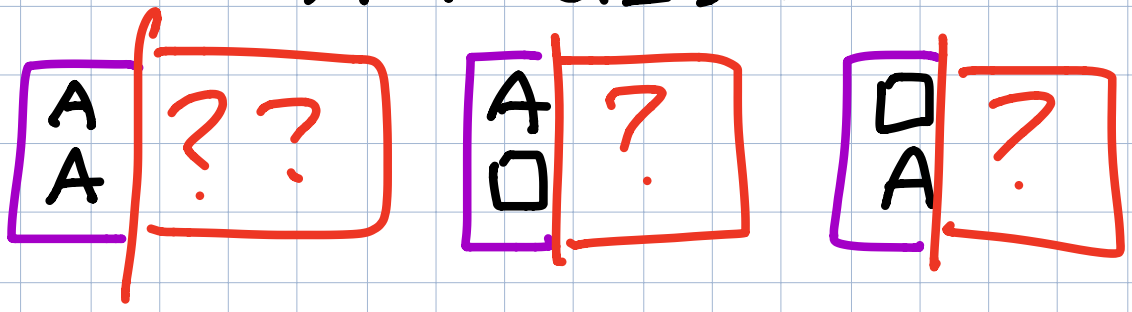
Repeated squaring $\rightarrow O(\log n)$ X's

Edit Distance

FOOD
MOOD
MOND
MONY
MONEY

Min # insertions deletions replacements to change A to B

ALGORITHMS
ALTRUISTIC



$$Edit(A[1..n], B[1..m]) =$$

$$\min \begin{cases} [A[1] \neq B[1]] + Edit(A[2..n] \\ \end{cases}$$

Subproblem:

ALGORI
ALTRUIST

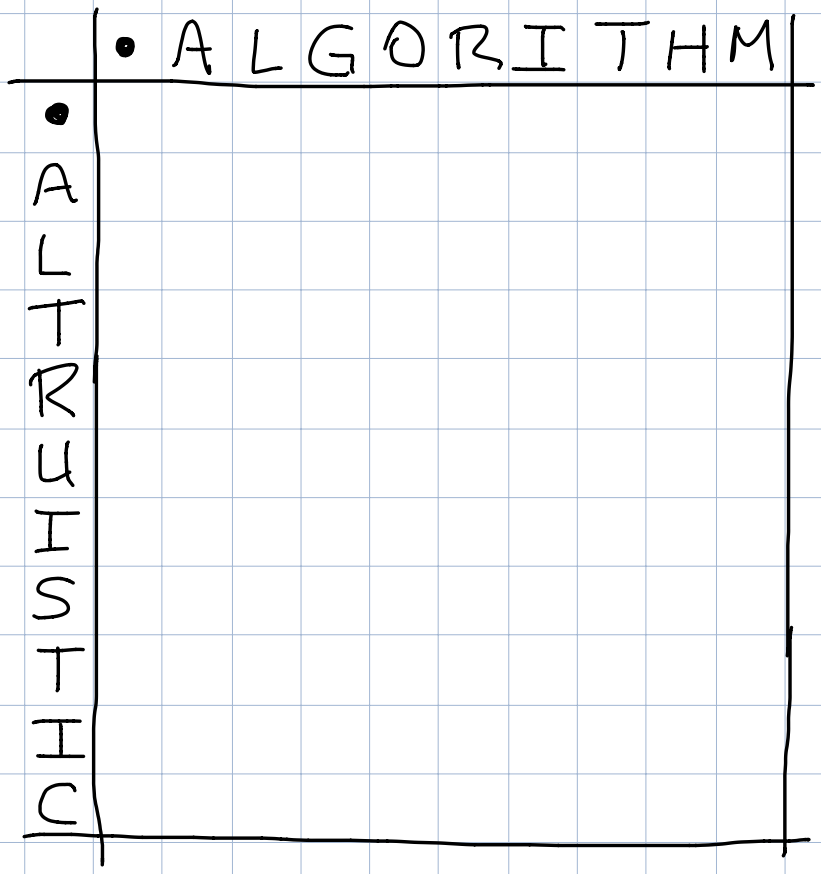
$$Edit(A[1..m], B[1..n]) = \min \left\{ \begin{array}{l} Edit(A[1..m-1], B[1..n]) + 1 \\ Edit(A[1..m], B[1..n-1]) + 1 \\ Edit(A[1..m-1], B[1..n-1]) + [A[m] \neq B[n]] \end{array} \right\}$$

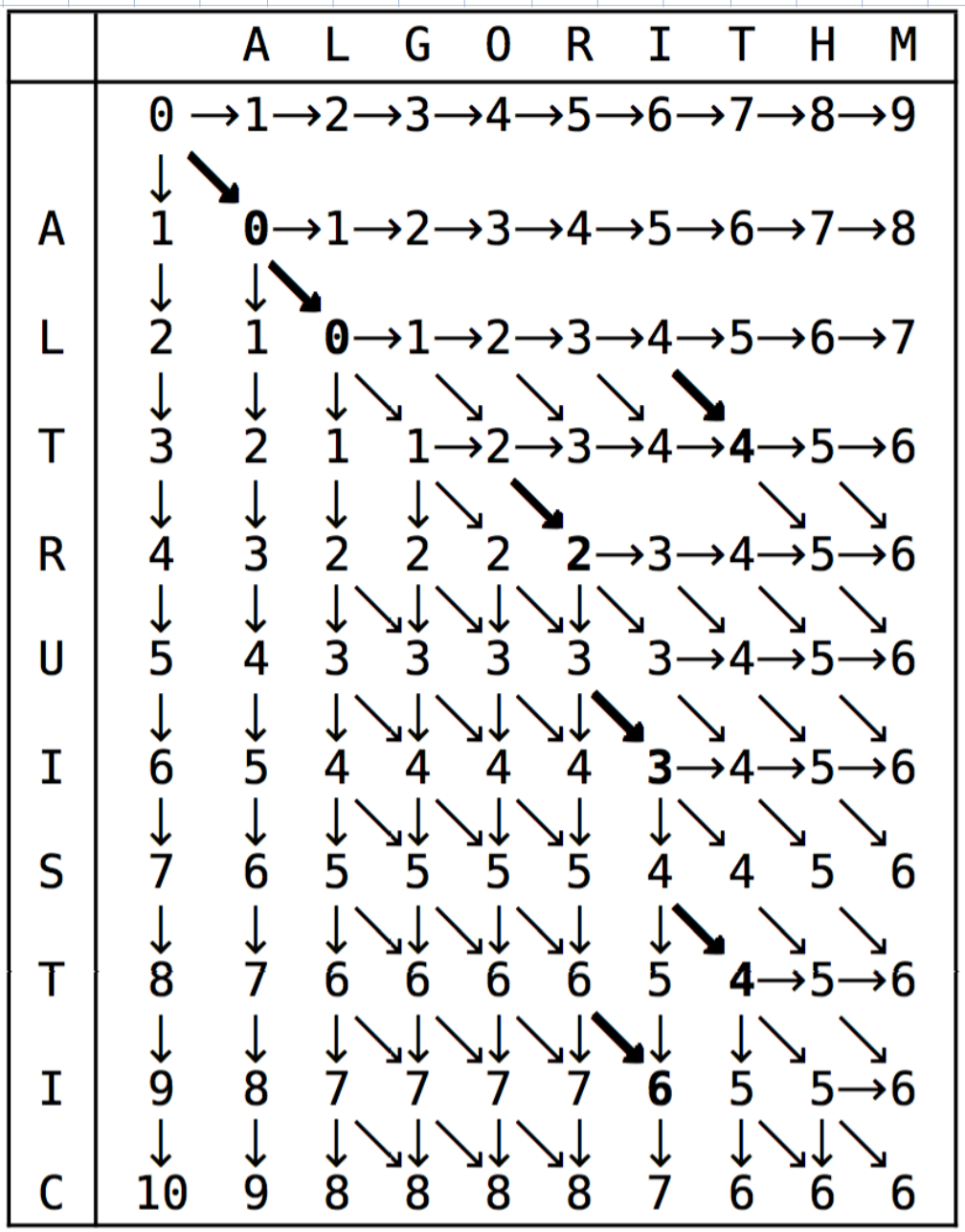
$$Edit(i,j) = \begin{cases} i & \text{if } j = 0 \\ j & \text{if } i = 0 \\ \min \left\{ \begin{array}{l} Edit(i-1,j)+1, \\ Edit(i,j-1)+1, \\ Edit(i-1,j-1)+\left[A[i] \neq B[j]\right] \end{array} \right. & \text{otherwise} \end{cases}$$

```
EditDistance(A[1..m], B[1..n]):
    for j ← 1 to n
        Edit[0, j] ← j

    for i ← 1 to m
        Edit[i, 0] ← i
        for j ← 1 to n
            if A[i] = B[j]
                Edit[i, j] ← min {Edit[i − 1, j] + 1, Edit[i, j − 1] + 1, Edit[i − 1, j − 1]}
            else
                Edit[i, j] ← min {Edit[i − 1, j] + 1, Edit[i, j − 1] + 1, Edit[i − 1, j − 1] + 1}

    return Edit[m, n]
```


• ALGORITHM
• ALTRUISTIC


A L G O R I T H M
0 1 2 3 4 5 6 7 8 9
A 1 0 1 2 3 4 5 6 7 8
L 2 1 0 1 2 3 4 5 6 7
T 3 2 1 1 2 3 4 4 5 6
R 4 3 2 2 2 2 3 4 5 6
U 5 4 3 3 3 3 3 4 5 6
I 6 5 4 4 4 4 3 4 5 6
S 7 6 5 5 5 5 4 4 5 6
T 8 7 6 6 6 6 5 4 5 6
I 9 8 7 7 7 7 6 5 5 6
C 10 9 8 8 8 8 7 6 6 6