Dynamic Programming

BY SASHA

What is DP?

- **Dynamic programming** is both a mathematical optimization method and a computer programming method. The method was developed by Richard Bellman in the 1950s and has found applications in numerous fields, from aerospace engineering to economics. In both contexts it refers to simplifying a complicated problem by breaking it down into simpler sub-problems in a recursive manner.

Why is dynamic programming called dynamic programming?

- Richard Bellman created dynamic programming to hide the fact that he was doing a mathematical research. Bellman was working at the place called Rand, and under a secretary of defense who had a pathological fear and hatred for the term research. So he settled on the term dynamic programming because it would be difficult to give a pejorative meaning to it. And because it was something not even a congressman could object to. Basically, it sounded cool.

DP = subproblem + “reuse”.

- A simple example to represent a concept.

1+1+1+1+1+1+1+1+1 = ?

And + 1 = ?

- Remember the past. So, you don’t have to repeat it

Example computing Fibonacci Numbers

- Fibonacci number

1, 1, 2, 3, 5, 8, 13, 21…

$F1 = F2 = 1$

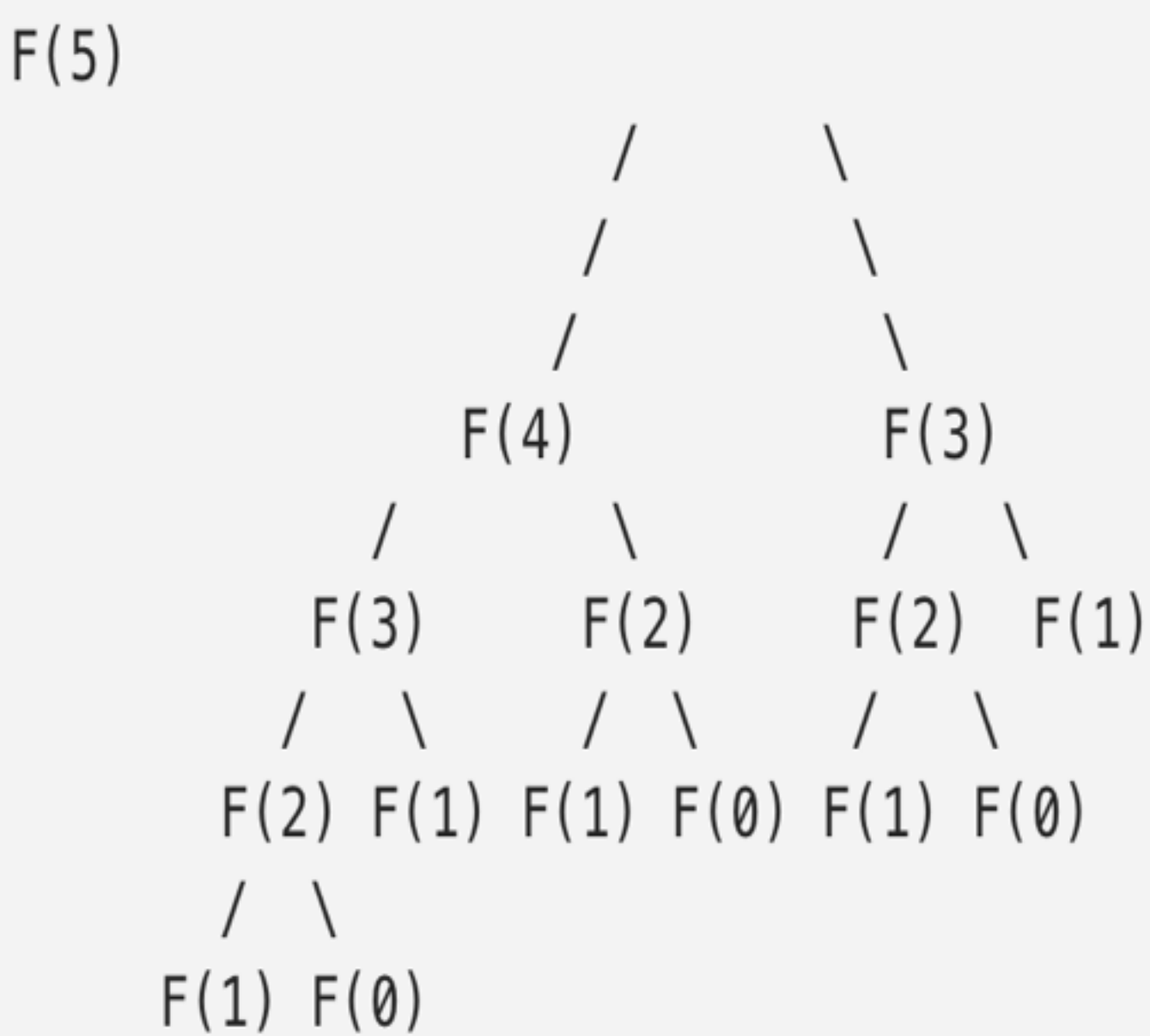
$Fn = Fn - 1 + Fn - 2$

Example computing Fibonacci Numbers

- Naïve recursive algorithm

```
fib(n) {

if n <=2 : f = 1
else f = fib(n-1) + fib(n-2)
return f;
}
```


F(5)
F(4)
F(3)
F(3)
F(2)
F(2)
F(1)
F(2)
F(1)
F(1)
F(0)
F(1)
F(0)
F(1)
F(0)

Memoizatin DP algorithm.

```
memo = []
Fib(n) :
If in memo: return memo[n]
If n <= 2: f = 1
Else: f = f(n-1)+ f(n-2)
Memo[n] = f
Return f
```

DP = recursion + memoization

- Memoize (remember) & reuse solutions to subproblems that help solve the problem.
- Time = number of subproblems * time spend per subproblem.

Bottom-up DP algorithm

```
public int fib(int n) {
    int[] output = new int[n + 1];
    output[1] = 1;
    output[2] = 1;
    for (int i = 3; i <= n; i++) {
    output[i] = output[i - 1] + output[i - 2];
}
  return output[n];
}
```

Now if we look into this algorithm it actually start from lower values then go to top. If i need 5th fibonacci number i am actually calculating 1st, then second then third all the way to up 5th number. This techniques actually called bottom-up techniques.

Recourses

- Dynamic programming – Wikipedia

https://en.wikipedia.org/wiki/Dynamic_programming

- MIT Open Course

https://www.youtube.com/watch?v=OQ5isbhAv_M

http://web.mit.edu/15.053/www/AMP-Chapter-11.pdf

- GeeksforGeeks

https://www.geeksforgeeks.org/tabulation-vs-memoization/