fibFuncDynProg

March 9, 2021

1 Dynamic Fibonacci Sequence

Function 'fib(n)' takes in an argument, returning the n-th number of Fibonacci sequence Recursive JavaScript Program is correct but inefficient.

```
const fib = (n) => {
  if (n <= 2) return 1;
  returnn fib(n - 1) + fib(n - 2);
};
console.log(fib(6));  // 8
console.log(fib(7));  // 13
console.log(fib(8));  // 21
console.log(fib(50)); // process hangs
```

Recursive Tree Diagram

```
          7
       6      5
     5   4   4   3
   4  3 3 2 3 2  21
  32 21 21 21
 21
```

Time Complexity

$$2(n)$$

```
const fib = (n, memo = {}) => {
  if (n in memo) return memo[n];
  if (n <= 2) return 1;
  memo[n] = fib(n - 1, memo) + fib(n - 2, memo);
  return memo[n];
};

// console.log(fib(50)); // 12586269025
```

Iterative Tree Diagram

```
6
```

```
  5   4
 4 3  3 2
32 21 21
21
```

Limiting number of recursive calls by passing in a 'memo' object];

$$2(n)$$

$$O(n)time$$

$$O(n)space$$

By 'memo-izing' our Fibonacci function, we brought the number of recursive calls down from exponential time to linear.