Actividad 9.1

```
if fibonacci (int n) {
    if(n==0) cond1
        return 0; ret1
    else if (n==1) cond2
        return 1; ret2
    else
        return fibonacci(n-1) + fibonacci(n-2)
}
```

$$T(n) = \begin{cases} n=0: & T(cond_1) + T(ret_1) = t + t = 2t \\ n=1: & T(cond_1) + T(cond_2) + T(ret_2) = t + t + t = 3t \\ n>1: & T(cond_1) + T(cond_2) + T(n-1) + T(n-2) = 3t + T(n-1) + T(n-2) \end{cases}$$

$$T(n) = 3t + \underline{T(n-1)} + \underline{T(n-2)}$$

$$= 3t + [3t + T(n-2) + T(n-3)] + [3t + T(n-3) + T(n-4)]$$

$$= 9t + T(n-2) + 2T(n-3) + T(n-4)$$

$$= 9t + [3t + T(n-3) + T(n-4) + 2] + 2[3t + T(n-4) + T(n-5)] + [3t + T(n-5) + T(n-6)]$$

$$= 24t + T(n-3) + 3T(n-4) + 3T(n-5) + T(n-6)$$

<u>No hay patrón</u>

$$T(n-1) > T(n-2) \rightarrow 3t + T(n-1) + T(n-1) > 3t + T(n-1) + T(n-2)$$

Resolviendo

$$T(n) = 3t + T(n-1) + T(n-1) = 3t + 2T(n-1)$$

$$= 3t + 2[3t + 2T(n-2)] = (1+2)3t + 2^2\,T(n-2)$$

$$= (1+2)3t + 2^2[3t + 2T(n-3)] = (1+2+3)3t + 2^3\,T(n-3)$$

$$= (1+2+3)3t + 2^3[3t + 2T(n-4)]$$

K-ésima expansión

$T(n) = (2^0, 2^1, 2^2, \ldots, + 2^{k-1}) 3t + 2^k (T(n-k))$

$T(0) = 2t \rightarrow n = k = 0$

$T(1) = 3t \qquad n = 1$

Sustituimos en (K)

$T(n) = (2^0, 2^1, 2^2, 2^3 \ldots \ldots 2^{k-1}) 3t + 2^n T(0)$

$= (2^0 + 2^1 + 2^2 + 2^{k-1}) 3t + 2^n (T(0))$

$T(n) = (2^0, 2^1, 2^2 \ldots \ldots 2^{k-1}) 3 + 2^n t1$

(2) $O(T(n)) = O((2^0, 2^1, 2^2 \ldots \ldots 2^{n-1}) 3 + 2^{n+1})$

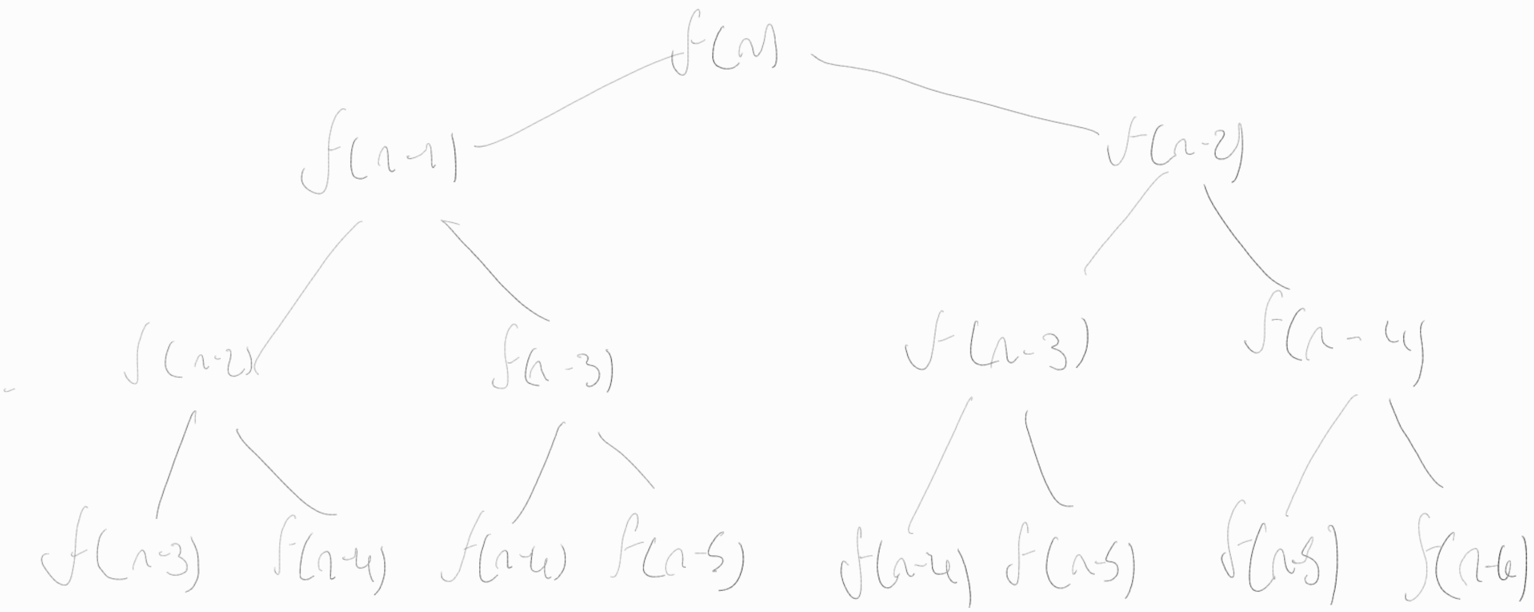
Regla de Suma

$= \max(O3(2^0), O3(2^1), O3(2^2) \ldots O3(2^{n-1}), O(2^{n+1}))$

$= O(2^{n+1})$

- Constantes

$O(T(n)) = O(n) = 2^n$



int fibonacci (int n){

③ int fib (int n, int &n1)

④ if (n==0){ cond1
return 0 ret1

else if (n==1){ cond2
n1=0 asig
return 1; ret2
} else{ int n2=0 Asig2
n1= fib(n-1, n2)
return n1+n2; ret2

$$T(n)\begin{cases} n=0 = T(cond_1) + T(ret_1) = t + t = 2t \\ n=1 = T(cond_1) + T(cond_2) + T(asig_1) + T(ret_2) = 4t \\ n>1.\ T(cond_1) + T(cond_2) + T(asig_2) + T(n-1) + T(ret_3) = 4t + T(n-1) \end{cases}$$

Expansión

$$T(n) = 4t + T(n-1)$$

$$= 4t + [4t + T(n-2)] = 2(4t) + T(n-2)$$

$$= 4t + [4t + T(n-3)] = (3)(4t) + T(n-3)$$

$$= 3(4t) + [4t + T(n-4)] = (4)\,4t + T(n-4)$$

K-ésima expansión

$$T(n) = (K)\,4t + T(n-k)$$

$$T(0) = 2t$$
$$T(1) = 4t$$
$$\rightarrow n-K=0 \rightarrow n=K$$

$$\rightarrow T(n) = n\,4t + T(0) = 4nt + T(0) = 4nt + 2t$$

$$\rightarrow T(n) = 4n+2$$

$$O(T(n)) = O(4n+2) = n$$

Estructuras de Datos

ANÁLISIS DE ALGORITMOS
A-1.1 Función de Fibonacci

René Ornelis
Vacaciones de junio de 2024

Javier Andrés Monjes Solózano
202100081

Optimización de función de Fibonacci

Objetivos

Los objetivos de esta actividad son que el estudiante sea capaz de:

1. Determinar y comprobar el orden de un algoritmo (función O(n))
2. Optimizar un algoritmo para lograr un mejor rendimiento

Problema

Consideremos el algoritmo de la función de Fibonacci:

```
int fibonacci(int n) {
        if (n <=0 )
                return 0 ;
        else if (n ==1)
                return 1 ;
        else
                return fibonacci(n-1) + fibonacci(n-2) ;
}
```

Este algoritmo tiene un orden $O(n) = n^2$, por lo que se requiere que cree una función recursiva equivalente (que dé el mismo resultado), pero que sea $O(n)=n$. No debe utilizar ninguna estructura de datos adicional.

Deberá entregar:

1. Demostración de que el algoritmo original es $O(n) = n^2$.
2. Algoritmo recursivo equivalente en C++.
3. Demostración que el algoritmo equivalente es $O(n)=n$.

1.Demostración de que el algoritmo original es O(n) = n^2

$$T(n) = \begin{cases} T(cond0) + T(ret0) = t + t = 2t & si \quad n \leq 0 \\ T(cond0) + T(cond1) + T(return1) = t + t + t = 3t & si \quad n = 1 \\ T(cond0) + T(cond1) + T(ret2) + T(n-1) + T(n-2) = 3t + T(n-1) + T(n-2) & si \quad n > 1 \end{cases}$$

$$T(n) = 3t + T(n-1) + T(n-2)$$

$$= 3t + [3t + T(n-2) + T(n-3)] + [3t + T(n-3) + T(n-4)] = (3*3)t + T(n-2) + 2T(n-3) + T(n-4)$$

$$= (3*3)t + [3t + T(n-3) + T(n-4)\ ] + 2[3t + T(n-4) + T(n-5)\ ] + [3t + T(n-5) + T(n-6)\ ]$$

$$= (7*3)t + T(n-3) + 3T(n-4) \ + 3T(n-5) + T(n-5)$$

$$= (7*3)t + [3t + T(n-4) + T(n-5)\ ] + 3[3t + T(n-5) + T(n-6)\ ] \ + 3[3t + T(n-6) + T(n-7)\ ] + [3t + T(n-6) + T(n-7)\ ]$$

$$= (15*3)t + T(n-4) + 4T(n-5) \ + 6T(n-6) + 4T(n-7) + T(n-8)$$

Sin Patrón aparente

Por aproximación

$$T(n-1) > T(n-2)$$

$$3t + T(n-1) + T(n-2) < 3t + T(n-1) + T(n-1)$$

$$3t + T(n-1) + T(n-2) < 3t + 2T(n-1)$$

$$T(n) = 3t + 2T(n-1)$$

$$= 3t + 2[3t + 2T(n-2)] = (3+3)t + 4T(n-2)$$

$$= (3+3)t + 4[3t + 2T(n-3)] = (6+6)t + 8T(n-3)$$

$$= (12)t + 8[3t + 2T(n-4)] = (12+12)t + 16T(n-4)$$

$$T(n) = \left(3x2^{k-1}\right)t + 2^k T(n-k)$$

$$Si\ T(0) = 2t$$

$$n - k = 0 \rightarrow \ k = n$$

$$T(n) = (3x2^{n-1})t + 2^n T(0) \rightarrow (3x2^{n-1})t + 2^n(2t) \rightarrow t(3x2^{n-1} + 2^n)$$

$$O\big(T(n)\big) = O(3x2^{n-1} + 2^n) = \max\big(O(3x2^{n-1}), O(2^n)\big) = O(2^n) = 2^n$$

$$\boldsymbol{O\big(T(n)\big) = 2^n}$$

2.

```
int fibonacci(int n, int a = 0, int b = 1) {
    if (n == 0) {
        return a;
    } else {
        return fibonacci(n - 1, b, a + b);
    }
}
```

3.

$$T(n) = \begin{cases} T(cond0) + T(ret0) = t + t = \ 2t\ 2t & si\ \ n \leq 0 \\ T(cond0) + T(ret1) + T(n-1) = 2t + T(n-1) & n > 0 \end{cases}$$

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

$$= 2t + [2t + T(n-2)\ ] = (4)t + T(n-2)$$

$$= 4t + [2t + T(n-3)] = T(n) = 6t + [T(n-3)]$$

$$= 6t + [2t + T(n-4)] = 8t + [T(n-4)]$$

$$T(n) = 2kt + T(n-k)$$

$$n - k = 0 \rightarrow k = n$$

$$T(n) = 2kt + T(n-k)$$

$$= nt + T(0) \rightarrow nt + 2t \rightarrow t(n+2)$$

$$O\big(T(n)\big) = O(n+2) = \max\big(O(n), O(2)\big) = O(n) = n$$

$$\boldsymbol{O\big(T(n)\big) = O(n)}$$