Análisis del algoritmo Fibonacci (5)

Código:

```
int fibo (int n) {
    if (n < 2) return 1;
① int v1 = fibo (n-1);
② int v2 = fibo (n-2);
    return v1 + v2;
③ }
```

n	v1	v2	Línea
5	?	?	1
4	?	?	1
3	?	?	1
2	?	?	1
1	?	?	3

n	v1	v2	Línea
5	?	?	1
4	?	?	1
3	?	?	1
2	1	?	2
0	?	?	3

n	v1	v2	Línea
5	?	?	1
4	?	?	1
3	?	?	1
2	1	1	3

n	v1	v2	Línea
5	?	?	1
4	?	?	1
3	2	?	2
1	?	?	3

n	v1	v2	Línea
5	?	?	1
4	?	?	1
3	2	1	3

n	v1	v2	Línea
5	?	?	1
4	3	?	2
2	?	?	1
1	?	?	3

n	v1	v2	Línea
5	?	?	1
4	3	?	2
2	1	?	2
0	?	?	3

n	v1	v2	Línea
5	?	?	1
4	3	?	2
2	1	1	3

n	v1	v2	Línea
5	?	?	1
4	3	2	3

n	v1	v2	Línea
5	?	?	2
3	?	?	1
2	?	?	1
1	?	?	3

n	v1	v2	Línea
5	5	?	2
3	?	?	1
2	1	?	2
0	?	?	3

n	v1	v2	Línea
5	5	?	2
3	?	?	1
2	1	1	3

n	v1	v2	Línea
5	5	?	2
3	2	?	2
1	?	?	3

n	v1	v2	Línea
5	5	?	2
3	2	1	3

n	v1	v2	Línea
5	5	3	3

return v1 + v2 : return 5 + 3 = return 8

n	v1	v2	Línea
$\emptyset$			

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

$T(1) = T(2) = 1$

Si partimos de la aproximación:

$T(n-1) + T(n-2) \leq 2T(n-1)$

Por tanto,

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

$T(1) = 1$

$$\left.\begin{array}{l} \text{Paso 1: } T(n) = 1 + 2T(n-1) \\ \text{Paso 2: } T(n) = 3 + 4T(n-2) \\ \text{Paso 3: } T(n) = 7 + 8T(n-3) \\ \quad \ldots \end{array}\right\} \text{Paso } K \Longrightarrow T(n) = (2^K - 1) + 2^K T(n-K)$$

Finaliza el caso base cuando $n - K = 1 \rightarrow K = n - 1$

Sustituimos: $T(n) = (2^{n-1} - 1) + 2^{n-1} T(n - n + 1) =$

$= (2^{n-1} - 1) + 2^{n-1} T(1) =$

$= 2^{n-1} - 1 + 2^{n-1} =$

$= \boxed{2^n - 1}$

Concluyendo:

$T(n) \in O(2^n - 1) = O(\max\{2^n, -1\}) = O(2^n)$