
ESCOM - IPN

Análisis Recursivo

ANÁLISIS DE ALGORITMOS 3CM3



Oscar Andrés Rosas Hernandez

Mayo 2018

Índice

1. Algoritmo 1	2
1.1. Código	2
1.2. Desarrollo	2
2. Algoritmo 2	3
2.1. Código	3
2.2. Desarrollo	3
3. Algoritmo 3	4
4. Algoritmo 3.1	4
5. Algoritmo 3.2	5
6. Algoritmo 3.3	6
7. Algoritmo 4	7
7.1. Código	7
7.2. Desarrollo	7
8. Algoritmo 5	8
8.1. Código	8
8.2. Desarrollo	8
9. Algoritmo 6	9
10. Algoritmo 6.1	9
11. Algoritmo 6.2	10

1. Algoritmo 1

1.1. Código

```
1 int FuncionRecursiva (int num) {  
2     if (num == 0)  
3         return 1;  
4     else if (num < 3) {  
5         resultado = 0;  
6         for(i=0; i < (num * num); i++)  
7             resultado += num;  
8         return resultado;  
9     }  
10    else  
11        return  
12        ((FuncionRecursiva(num - 1) * FuncionRecursiva(num - 2)) / FuncionRecursiva(num - 3));  
13 }
```

1.2. Desarrollo

Este lo podemos caracterizar por la forma:

- $T(0) = 1$
- $T(1) = 3$
- $T(2) = 6$
- $T(3) = 9$
- $T(n) = T(n-1) + T(n-2) + T(n-3) \quad \forall n \geq 3$

Pero antes vamos a transformarlo a su ecuación característica:

$$x^3 - x^2 - x - 1 = 0$$

Sus raíces son:

- $x_1 \approx 1.83928675$
- $x_2 \approx -0.41964 - 0.60629i$
- $x_3 \approx -0.41964 + 0.60629i$

Entonces tenemos que:

$$T(n) = c_1(1.83928675)^n + c_2(-0.41964 - 0.60629i)^n + c_3(-0.41964 + 0.60629i)^n$$

Ahora vamos a calcular las constantes: $T(1) = 1 = c_1(1.83928675) + c_2(-0.41964) + c_3(-0.41964)$ $T(2) = 3 = c_1(3.382) + c_2(-0.191490) + c_3(-0.191490)$ $T(3) = 9 = c_1(6.222) + c_2(-0.073) + c_3(-0.073)$

Teniendo que $c_3 \neq 0$, entonces tenemos que $O(k^n)$

2. Algoritmo 2

2.1. Código

```
1 int FuncionRecursiva (int num) {  
2     if (num == 0)  
3         return 1;  
4     else if (num < 3) {  
5         resultado = 0;  
6         for(i=0; i < (num * num); i++)  
7             resultado += num;  
8         return resultado;  
9     }  
10    else  
11        return  
12        ((FuncionRecursiva(num - 1) * FuncionRecursiva(num - 2)) / FuncionRecursiva(num - 3));  
13 }
```

2.2. Desarrollo

Este lo podemos caracterizar por la forma:

- $T(0) = 1$
- $T(1) = 1$
- $T(2) = 1$
- $T(3) = 3$
- $T(n) = T(n-1) + T(n-2) + T(n-3) \quad \forall n \geq 3$

Pero antes vamos a transformarlo a su ecuación característica:

$$x^3 - x^2 - x - 1 = 0$$

Sus raíces son:

- $x_1 \approx 1.83928675$
- $x_2 \approx -0.41964 - 0.60629i$
- $x_3 \approx -0.41964 + 0.60629i$

Entonces tenemos que:

$$T(n) = c_1(1.83928675)^n + c_2(-0.41964 - 0.60629i)^n + c_3(-0.41964 + 0.60629i)^n$$

Ahora vamos a calcular las constantes:

3. Algoritmo 3

4. Algoritmo 3.1

- $T(0) = 0$
- $T(1) = 1$
- $T(n) = 3T(n-1) + 4T(n-2) \quad \forall n \geq 2$

Pero antes vamos a transformarlo a su ecuación característica:

$$x^2 - 3x - 4 = 0$$

Las raíces son:

- $x = -1$
- $x = 4$

Entonces tenemos que:

$$T(n) = c_1(-1)^n + c_24^n$$

Ahora vamos a calcular las constantes:

- $T(0) = 0 = c_1 + c_2$
- $T(1) = 1 = -c_1 + 4c_2$

Entonces es claro que las constantes valen:

- $c_1 = -\frac{1}{5}$
- $c_2 = \frac{1}{5}$

Es decir tenemos la función complejidad tiene un valor de $T(n) = -\frac{1}{5}(-1)^n + \frac{1}{5}(4)^n$

Es decir es de orden $O(4^n)$

5. Algoritmo 3.2

- $T(0) = 5$
- $T(1) = 27$
- $T(2) = 129 = 3(27) + 4(5) + (7)4$
- $T(3) = 559 = 3(129) + 4(27) + (8)8$
- $T(n) = 3T(n-1) + 4T(n-2) + (n+5)2^n \quad \forall n \geq 2$

Pero antes vamos a transformarlo a su ecuación característica:

$$(x^2 - 3x - 4)(x - 2)^2 = 0(x + 1)(x - 4)(x - 2)^2 = 0$$

Las raíces son:

- $x = -1$
- $x = 4$
- $x = 2$
- $x = 2$

Entonces tenemos que:

$$T(n) = c_1(-1)^n + c_24^n + c_3(2)^n + c_4n(2)^n$$

Ahora vamos a calcular las constantes:

- $T(0) = 5 = c_1 + c_2 + c_3$
- $T(1) = 27 = -c_1 + 4c_2 + 2c_3 + 2c_4$
- $T(2) = 129 = c_1 + 16c_2 + 16c_3 + 8c_4$
- $T(3) = 559 = -c_1 + 64c_2 + 64c_3 + 192c_4$

Ninguna de las constantes son cero, por lo tanto es obvio que $O(2n^n)$

6. Algoritmo 3.3

- $T(0) = 5$
- $T(1) = 1$
- $T(n) = 2T(n-1) + (1)3^n \quad \forall n \geq 2$

Pero antes vamos a transformarlo a su ecuación característica:

$$(x-2)(x-3) = 0$$

Las raíces son:

- $x = 2$
- $x = 3$

Entonces tenemos que:

$$T(n) = c_1(2)^n + c_2(3)^n$$

Ahora vamos a calcular las constantes:

- $T(0) = 0 = c_1 + c_2$
- $T(1) = 1 = 2c_1 + 3c_2$

Entonces es claro que las constantes valen:

- $c_1 = -1$
- $c_2 = 1$

Es decir tenemos la función complejidad tiene un valor de $T(n) = (-1)2^n + 3^n$

Es decir es de orden $O(3^n)$

7. Algoritmo 4

7.1. Código

```
1 int BusquedaBinaria(int num_buscado, int numeros[], int inicio, int centro, int final) {  
2     if (inicio > final)  
3         return -1;  
4     else if (num_buscado == numeros[centro])  
5         return centro;  
6     else if (num_buscado < numeros[centro])  
7         return BusquedaBinaria(num_buscado, numeros, inicio, (int)((inicio+centro-1)/2), centro-1);  
8     else  
9         return BusquedaBinaria(num_buscado, numeros, centro+1, (int)((final+centro+1)/2), final);  
10 }
```

7.2. Desarrollo

Este lo podemos caracterizar por la forma:

- $T(0) = 0$
- $T(n) = 3 + T(n/2)$

Ahora vamos a usar el teorema maestro tenemos que $f(n) = 3$, $b = 2$, $a = 1$ y vemos claramente que:

$$f(n) = 3 = O(1) = O(n^{\log_2 1}) = O(n^0).$$

Por lo tanto es sencillo ver que $T(n) \in O(\log_2(n))$

8. Algoritmo 5

8.1. Código

```
1 MergeSort(a, p, r)
2 {
3     if ( p < r )
4     {
5         q = parteEntera((p+r)/2);
6         MergeSort(a, p, q);
7         MergeSort(a, q+1, r);
8         Merge(a, p, q, r);
9     }
10 }
```

8.2. Desarrollo

Este lo podemos caracterizar por la forma:

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

Ahora vamos a usar el teorema maestro tenemos que $f(n) = n$, $b = 2$, $a = 2$ y vemos claramente que:

$$f(n) = n = O(n) = O(n^{\log_2 2}) = O(n^1).$$

Por lo tanto es sencillo ver que $T(n) \in O(n \ln(n))$

9. Algoritmo 6

10. Algoritmo 6.1

$$T(n) = 3T(n/3) + 4(n/2) + 2n^2 + n$$

Ahora vamos a usar el teorema maestro tenemos dividiendo en 2 partes:

- $3T(n/3) + n$

Ahora vamos a usar el teorema maestro tenemos que $f(n) = n$, $b = 3$, $a = 3$ y vemos claramente que:

$$f(n) = n = O(n) = O(n^{\log_3 3}) = O(n^1).$$

Por lo tanto es sencillo ver que $T(n) \in O(n \ln(n))$

- $4T(n/2) + 2n^2$

Ahora vamos a usar el teorema maestro tenemos que $f(n) = 2n^2$, $b = 2$, $a = 4$ y vemos claramente que:

$$f(n) = 2n^2 = O(n^2) = O(n^{\log_2 4}) = O(n^2).$$

Por lo tanto es sencillo ver que $T(n) \in O(n^2 \ln(n))$

Por lo tanto el algoritmo en general se da en $O(n^2 \ln(n))$

11. Algoritmo 6.2

$$T(n) = 3T(n/3) + 4(n/2) + 2n^2 + n$$

Ahora vamos a usar el teorema maestro tenemos dividiendo en 2 partes:

■ $3T(n/3) + n$

Ahora vamos a usar el teorema maestro tenemos que $f(n) = n$, $b = 3$, $a = 3$ y vemos claramente que:

$$f(n) = n = O(n) = O(n^{\log_3 3}) = O(n^1).$$

Por lo tanto es sencillo ver que $T(n) \in O(n \ln(n))$

■ $4T(n/2) + 2n^2$

Ahora vamos a usar el teorema maestro tenemos que $f(n) = 2n^2$, $b = 2$, $a = 4$ y vemos claramente que:

$$f(n) = 2n^2 = O(n^2) = O(n^{\log_2 4}) = O(n^2).$$

Por lo tanto es sencillo ver que $T(n) \in O(n^2 \ln(n))$

Por lo tanto el algoritmo en general se da en $O(n^2 \ln(n))$