

Apuntes-Tema-1.pdf



Juandf03



Análisis y diseño de algoritmos



2º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

Formamos
talento para un futuro
Sostenible

EOI Escuela de
organización
industrial



MÁSTER EN
**Big Data &
Business Analytics**

[saber más](#)



PC WORK: TU ALIADO

PERFECTO PARA LA UNI

Encuentra tu NitroPC ideal para edición, renderizado y trabajos de alto rendimiento, diseñado para oficina y estudio.



Tema 1 "Complejidad de algoritmos"

1 Coste computacional de un algoritmo

Ej! 2^n , $n = 32$

- Método directo $\rightarrow 32 \text{ mult}$
- Método eficiente $\rightarrow 5 \text{ mult}$

$\text{Coste eficiente} = \log_2 (\text{Coste directo})$

• Complejidad según el tamaño de la entrada

función TA: IN \rightarrow IN asociar a cada entrada su tiempo de ejecución

• Reglas para hallar el coste de un algoritmo:

→ Instrucciones simples (operaciones elementales) **COSTE 1**
se ejecutan en tiempo constante $i=3$; $i>k$

→ Composición de instrucciones $T_{I_1, I_2}(n) = T_{I_1}(n) + T_{I_2}(n)$
 $i=0$ y $i > n$ // $i++$

→ Instrucciones de selección $T_{\text{selección}}(n) = T_{\text{condición}}(n) + \max(T_1(n), T_2(n))$
se al coste mayor (if , switch, ...)

→ Bucle con iteraciones fijas **coste constante** ?
Ej! $for (int i=0; i<k; i++)$ // bloque con conste B

$T(n) = \text{Inicialización} + \left(\sum_{i=0}^{k-1} \text{comprobación} + \text{Iteraz} + B \right) + \text{comprobación}$

n° ijo de veces que se repite el bucle

$= 1 + \left(\sum_{i=0}^{k-1} 1 + 2 + B \right) + 1 = 1 + (3+B) \cdot k + 1$





PC GAMING: RINDE AL MÁXIMO EN TUS JUEGOS

Descubre equipos de alto rendimiento tanto para jugadores ocasionales como expertos. ¡Encuentra tu Nitropc ideal!



Escanea el Qr y encuentra el PC
perfecto para ti en www.nitro-pc.es



Análisis y diseño de algoritmos



Comparte estos flyers en tu clase y consigue más dinero y recompensas

- 1 Imprime esta hoja
- 2 Recorta por la mitad
- 3 Coloca en un lugar visible para que tus compis puedan escanear y acceder a apuntes
- 4 Llévate dinero por cada descarga de los documentos descargados a través de tu QR



Banco de apuntes de la



2. Clases de complejidad y notación asintótica !

- Tipos de complejidad: órdenes de crecimiento

$$n \text{ (entrada)} \Rightarrow \begin{array}{c} \text{c. logarítmica} \\ \log_2(n) \end{array} \Rightarrow \begin{array}{c} \text{c. lineal} \\ n \end{array} \Rightarrow \begin{array}{c} \text{c. cuadrática} \\ n^2 \end{array} \Rightarrow \begin{array}{c} \text{c. exponencial} \\ 2^n \end{array}$$

$$1 \ll \log n \ll n \ll n \log n \ll n^2 \ll \dots \ll 2^n \ll n!$$

- Notación asintótica: $O(g(n)) = \{f: \mathbb{N} \rightarrow \mathbb{N} / \exists k > 0, c > 0, \forall n > k, f(n) \leq c \cdot g(n)\}$

$$n \in O(n^2) \quad 100n + 5 \in O(n^2) \quad n^4 + n + 1 \notin O(n^2)$$

conjunto de funciones acotadas superiormente por un múltiplo de g

- Notación asintótica: $\Omega(g(n)) = \{f: \mathbb{N} \rightarrow \mathbb{N} / \exists k > 0, c > 0, \forall n > k, f(n) \geq c \cdot g(n)\}$

$$n^3 \in \Omega(n^2) \quad 100n + 5 \in \Omega(n) \quad n \notin \Omega(n^2)$$

conjunto de funciones acotadas inferiormente por un múltiplo de g

- Notación asintótica: $\Theta(g(n)) = \{f: \mathbb{N} \rightarrow \mathbb{N} / \exists k > 0, c_1 > 0, c_2 > 0 \dots\}$

$$100n^2 \in \Theta(n^2) \quad 100n + 5 \in \Theta(n) \quad \text{acotado exacto}$$

• Propiedades

$$\boxed{\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)}} = \begin{cases} 0 & \longrightarrow f(n) \in O(g(n)) \\ c & \longrightarrow f(n) \in \Theta(g(n)) \\ \infty & \longrightarrow f(n) \in \Omega(g(n)) \end{cases}$$

3. Algoritmos no recursivos : algoritmos de ordenación

$$\sum c_i \cdot a_i = c \cdot \sum a_i$$

$$\sum a_i + b_i = \sum a_i + \sum b_i$$

$$\sum_{i=\min}^{\max} i = \max - \min + 1$$

$$\sum_{i=0}^n i = \dots = \frac{n(n+1)}{2} \in \Theta(n^2)$$

- O. por inserción

15!

```
public static void insercion (int [] a) {  
    for (int i = 1; i < a.length; i++) {
```

int j = i - 1;

int x = a[i];

while (j >= 0 && a[j] > x) {

a[j + 1] = a[j];

j--;

a[j + 1] = x;

}

⑤ Caso medio $\Theta(n^2)$

① Tamaño en base : long array, n

② Op. elementales : asignación y comparación

③ Caso mejor : array ordenado

$$C(n) = 3(n-1) \in \underline{\Theta(n)}$$

④ Caso peor : array desordenado

$$C(n) = \sum_{i=1}^{n-1} \left(3 + \sum_{j=i+1}^n 2 \right) = \\ \dots = n^2 + 2n - 3 \in \underline{\Theta(n^2)}$$

- O. por selección \Rightarrow complejidad n^2

- O. por el método de la Burbuja \Rightarrow complejidad n^2



PC GAMING: RINDE AL MÁXIMO EN TUS JUEGOS

Descubre equipos de alto rendimiento tanto para jugadores ocasionales como expertos. ¡Encuentra tu NitroPC ideal!



4. Algoritmos Recursivos: Ecuaciones de recurrencia

- Factorial (algoritmo se llama a sí mismo)

$$M(n) = \underbrace{M(n-1)}_{\text{multiplicaciones para calcular factorial } (n-1)} + 1 ; n > 0$$

$M(0) = 0$

caso fácil (base)

SUSTITUCIÓN HACIA ATRÁS !

$$M(n) = M(n-1) + 1 = M(n-2) + 2 = \dots = M(n-n) + n = n$$

POLINOMIO CARACTERÍSTICO !

El tiempo de ejecución no puede ser negativo

① Ecuación homogénea $a_0 M(n) + a_1 M(n-1) + \dots + a_k M(n-k) = 0$

② Determinar Polinomio Característico $a_0 z^k + a_1 z^{k-1} + \dots + a_k = 0$

③ Raíces y multiplicidad

④ $M(n) = (A_n + B_n) z^n + C(z)^n$

multiplicidad raíz

caso bases y sistema
 $M(n) = \boxed{\quad}$

Ej! $M(n) = M(n-1) + M(n-2)$

$$M(n) - M(n-1) - M(n-2) = 0 \implies x^2 - x - 1 = 0$$

$$x_1 = \frac{1-\sqrt{5}}{2} \qquad x_2 = \frac{1+\sqrt{5}}{2}$$

$$M(n) = \left(\frac{1-\sqrt{5}}{2}\right)^n \cdot A + \left(\frac{1+\sqrt{5}}{2}\right)^n \cdot B$$

Orden de complejidad \rightarrow grado polinomio

Grado 0

$$M(n) - 2M(n-1) + M(n-2) = 0 ; \quad z^2 - 2z + 1 = 0 \implies z=1 \quad m=2$$

$$1^n (A_n + B_n) = 0 ; \quad A_n + B_n = 0$$

$$M(n) = n!^n$$

$$\left. \begin{array}{l} M(0) = 0 \\ M(1) = 1 \end{array} \right\} ; \quad \boxed{0 = B}$$

$$\left. \begin{array}{l} 1 = A + B \\ A = 1 \end{array} \right\}$$



① Ecuación no Homogénea $a_0 M(n) + a_1 M(n-1) + \dots + a_k M(n-k) = p(n) \cdot b^n$

② Polinomio característico $(z^k + a_1 z^{k-1} + \dots + a_k) \cdot (z - b)^{e+1} = 0$

...

grado del polinomio que acompaña a b^n

CUIDADO! $\left\{ \begin{array}{l} T(n) = 2T(n-1) + 4T(n-2) + 3^{2n} (n+5) \\ (x^2 - 2x - 4)(x-9)^2 = 0 \end{array} \right.$

$$3^{2n} = (3^2)^n$$

• Cambio de variable

Ejemplo

$$A(n) = 2A\left(\frac{n}{2}\right) + 1$$

Cambio $\rightarrow n = z^k$

$$A(z^k) = 2A(z^{k-1}) + 1$$

$$\bullet A(z^k) = F(k)$$

$$\bullet A(z^{k-1}) = F(k-1)$$

$$F(k) = zF(k-1) + 1$$

Resuelve por el polinomio característico

$$(x-2)(x-1) = 0$$

$$F(k) = \alpha_1 \cdot z^k + \alpha_2 \cdot 1^k$$

$$F(0) = A(z^0) = A(1) = \boxed{0}$$

$$F(1) = zF(0) + 1 = \boxed{1}$$

$$\left\{ \begin{array}{l} F(0) = \alpha_1 + \alpha_2 = 0 \\ F(1) = z\alpha_1 + \alpha_2 = 1 \end{array} \right.$$

$$\begin{cases} \alpha_1 = 1 \\ \alpha_2 = -1 \end{cases}$$

$$\Rightarrow F(k) = z^k - 1$$

$$A(z^k) = z^k - 1$$

$$\boxed{A(n) = n - 1 \in \Theta(n)}$$

Deshacemos el cambio

Solución

5. Teorema maestro

Da el orden de complejidad del algoritmo sin llegar a conocer la expresión.

→ Versión general

$$T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)$$

nº divisiones del problema tamaño divisiones

esfuerzo

$$T(n) \in \begin{cases} \Theta(n^{\log_b a}) & \text{si } f(n) \in O(n^{\log_b a} - \epsilon) \\ \Theta(f(n)) & \text{si } f(n) \in \Omega(n^{\log_b a} + \epsilon) \\ \Theta(n^{\log_b a} \log^{k+1} n) & \text{si } f(n) \in \Theta(n^{\log_b a} \log^k n) \end{cases}$$

Ej 1!

$$f(n) = n^d$$

$$n^d \in O(n^{\log_b a} - \epsilon)$$

$$\{ d < \log_b a - \epsilon \}$$

Ej 2!

$$f(n) = n^d$$

$$n^d \in \Omega(n^{\log_b a} + \epsilon)$$

$$\{ d > \log_b a \}$$

$$\exists \epsilon < 1, \exists n_0 > 0$$

$$\forall n > n_0$$

$$a \cdot f\left(\frac{n}{b}\right) \leq e f(n)$$

$$T(n) = 9T\left(\frac{n}{3}\right) + n$$

$$T(n) = 9T\left(\frac{n}{3}\right) + n^3$$

$$\log_b a = \log_3 9 = 2$$

$$\log_b a = \log_3 9 = 2$$

$$d = 2 \Rightarrow 2 < 2$$

$$d = 3 \Rightarrow 3 > 2$$

$$T(n) \in \Theta(n^2)$$

$$T(n) \in \Theta(n^3)$$

Ej 3!

$$f(n) = n^d$$

$$n^d \in \Theta(n^{\log_b a})$$

$$\{ d = \log_b a \}$$

$$T(n) = 9T\left(\frac{n}{3}\right) + n^2$$

$$\log_b a = \log_3 9 = 2$$

$$d = 2 \Rightarrow 2 = 2$$

$$T(n) = \Theta(n^2 \log n)$$

→ Versión reducida ($f(n)$ es fijo d pq es un polinomio)

$$T(n) \in \begin{cases} \Theta(n^{\log_b a}) & \text{si } a > b^d \\ \Theta(n^d) & \text{si } a < b^d \\ \Theta(n^d \log n) & \text{si } a = b^d \end{cases}$$

15! $T(n) = 2T\left(\frac{n}{2}\right) + 1$; $f(n) = 1 \Rightarrow d = 0$

$$\left. \begin{array}{l} a=2 \\ b=z \end{array} \right\} z > z^0, z > 1 \Rightarrow T(n) \in \Theta(n^{\log_z 2}) = \Theta(n)$$

15! **Importante** $T(n) = 3T\left(\frac{n}{4}\right) + n \log n$

$$a=3, b=4, f(n) = n \log n \Rightarrow d = 1 + \varepsilon$$

$$\log_b a = \log_4 3 \rightarrow \text{grado} < 1 \quad \text{0,792...}$$

- Estudiamos las posibilidades :

1) $n \log n \in O(n^{\log_4 3 - \varepsilon})$ [Falso]

2) $n \log n \in \Omega(n^{\log_4 3 + \varepsilon})$ [Verdadero] $\Rightarrow T(n) \in \Theta(n \log n)$

- Obtener α e (condición de regularidad) : $\exists \varepsilon < 1$

$$\alpha \cdot f\left(\frac{n}{b}\right) \leq e \cdot f(n) \quad ; \quad 3 \cdot f\left(\frac{n}{4}\right) \leq e \cdot (n \log n) \quad ;$$

$$3 \cdot \left(\frac{n}{4} \cdot \log \frac{n}{4}\right) \leq e \cdot (n \cdot \log n) \quad ;$$

$$\frac{3}{4} \cdot \cancel{e} \cdot \log \frac{n}{4} \leq e \cdot \cancel{e} \cdot \log n \quad ; \quad \frac{3}{4} (\log n - z) \leq e \cdot \log n$$

$$e = \frac{3}{4} < 1$$



PC GAMING: RINDE AL MÁXIMO EN TUS JUEGOS

Descubre equipos de alto rendimiento tanto para jugadores ocasionales como expertos. ¡Encuentra tu Nitropc ideal!



→ Versión reducida 2.0 Ta Maestro !

$$\log_b a > d \longrightarrow T(n) \in \Theta(n^{\log_b a})$$

$$\log_b a < d \longrightarrow T(n) \in \Theta(n^d)$$

$$\log_b a = d \longrightarrow T(n) \in \Theta(n^d \log n)$$

(Ej) Resuelve y orden de complejidad

$$T(n) = 4T(n/2) + n \quad T(1) = 1 \quad T(2) = c$$

1) Cambio variable: $n = 2^k$, $k = \log_2 n$

$$T(2^k) = 4T(2^{k-1}) + 2^k$$

$$F(k) = 4F(k-1) + 2^k \rightarrow \text{Forma ec. recurrencia básica}$$

$$\left\{ \begin{array}{l} F(k) = T(2^k) \\ F(k-1) = T(2^{k-1}) \end{array} \right.$$

2) Resolvemos ec. recurrencia:

$$(x-4)(x-2) = 0 \rightarrow \boxed{x_1 = 4} \quad \boxed{x_2 = 2}$$

$$F(k) = \alpha_1 \cdot 4^k + \alpha_2 \cdot 2^k$$

Deshacemos el cambio, $T(n)$

3) Deshacemos el cambio: $T(n) = \alpha_1 \cdot n^2 + \alpha_2 \cdot n$

sacamos las soluciones

$$\begin{cases} \alpha_1 + \alpha_2 = 1 \\ \alpha_1 \cdot 4 + \alpha_2 \cdot 2 = 6 \end{cases} \Rightarrow \boxed{\alpha_1 = 2} \quad \boxed{\alpha_2 = 1}$$

Obtenemos α_1 y α_2
después de deshacer el cambio de
variable

4) Complejidad: $T(n) \in \Theta(n^2)$



15) Resolver ecuación $T(n) = 2T\left(\frac{n}{4}\right) + n^{1/2}$ $T(1) = 1$

1) $n = 4^k$, $k = \log_4 n$

$$T(4^k) = 2T(4^{k-1}) + (4^k)^{1/2}$$

$$\begin{cases} T(4^k) = F(k) \\ T(4^{k-1}) = F(k-1) \end{cases}$$

$F(k) = 2F(k-1) + 2^k$

2) $(x-2)(x-2) = 0 \Rightarrow z_1 = 2, m_1 = 2 \Rightarrow F(k) = \alpha_1 z^k + \alpha_2 k z^k$

$$\begin{cases} T(4^0) = F(0) \\ T(4^0) = T(1) = 1 \end{cases} \Rightarrow F(0) = 1$$

$$\begin{cases} F(1) = 2F(0) + 2 \\ F(1) = 2 \cdot 1 + 2 = 4 \end{cases} \Rightarrow F(1) = 4$$

$$\begin{cases} \alpha_1 = 1 \\ 2\alpha_1 + 2\alpha_2 = 4 \\ \alpha_2 = 1 \end{cases}$$

$$F(k) = z^k + kz^k \quad T(4^k) = z^k + kz^k$$

3) $z^k = \sqrt{n}$

$T(n) = \sqrt{n} + \log_4 n \cdot \sqrt{n}$

PROPIEDADES DE LOS LOGARITMOS

1. $\log_a 1 = 0$

5. $\log_a x/y = \log_a x - \log_a y$

2. $\log_a a = 1$

6. $a^{\log_b x} = x^{\log_b a}$ | x suele ser una n |

3. $\log_a x^y = y \log_a x$

7. $\log_a x = \frac{\log_b x}{\log_b a} = \log_a b \log_b x$

4. $\log_a xy = \log_a x + \log_a y$