

Algoritmos y Estructuras de Datos II - 1º cuatrimestre 2015

Práctico 1 - Parte 3

Objetivo: adquirir buen manejo algebraico de la notación \mathcal{O} (ejercicios 1, 2 y 3), descubrir algunas de sus propiedades (4 y 5), familiarizarse con las notaciones auxiliares Ω y Θ (6 y 7), practicar la resolución sistemática de recurrencias (8, 10 y 11) y también las artesanales (9). Análisis de algunos algoritmos no elementales (12 y 13).

Dificultad estimada: Primer nivel de dificultad: 8 y 10. Segundo: 2 y 3. Tercero: 6 y 7. Cuarto: 1, 4 y 5. Quinto: 11, 12 y 13. Sexto: 9.

1. Determiná cuáles de las siguientes afirmaciones son verdades y cuáles falsas. Justificá apropiadamente.

- (a) $\mathcal{O}(f + g) = \mathcal{O}(\max(f, g))$.
- (b) Si $s \in \mathcal{O}(f)$ y $r \in \mathcal{O}(g)$ entonces $s + r \in \mathcal{O}(f + g)$.
- (c) Si $s \in \mathcal{O}(f)$ y $r \in \mathcal{O}(g)$ entonces $s - r \in \mathcal{O}(f - g)$.
- (d) $2^{n+1} \in \mathcal{O}(2^n)$.
- (e) $(m + 1)! \in \mathcal{O}(m!)$.

2. Ordená utilizando \subset e $=$ los órdenes (\mathcal{O}) de las siguientes funciones. No calcules límites, utilizá las propiedades algebraicas.

$$n \log 2^n \qquad 2^n \log n \qquad n! \log n \qquad 2^n$$

3. Determiná la relación (\subset , $=$) entre los órdenes (\mathcal{O}) de las siguientes funciones. Justificá sin utilizar la regla del límite.

$$n^4 + 2 \log n \qquad \log(n^{n^4}) \qquad 2^{4 \log n} \qquad 4^n \qquad n^3 \log n$$

4. Si $\lim_{n \rightarrow \infty} h(n) = \infty$, ¿cuáles de las siguientes afirmaciones resultan verdaderas?

- (a) si $f(n) \in \mathcal{O}(g(n))$, entonces $f(h(n)) \in \mathcal{O}(g(h(n)))$,
- (b) si $f(h(n)) \in \mathcal{O}(g(h(n)))$, entonces $f(n) \in \mathcal{O}(g(n))$,
- (c) si $\mathcal{O}(f(n)) \subseteq \mathcal{O}(g(n))$, entonces $\mathcal{O}(f(h(n))) \subseteq \mathcal{O}(g(h(n)))$
- (d) si $\mathcal{O}(f(n)) \subset \mathcal{O}(g(n))$, entonces $\mathcal{O}(f(h(n))) \subset \mathcal{O}(g(h(n)))$
- (e) si $f(n) \in \mathcal{O}(g(n))$, entonces $h(f(n)) \in \mathcal{O}(h(g(n)))$.

5. ¿Cuál de estas dos funciones crece más rápido: $\log n!$ ó $n \log n$? ¿y $\log(n^n)$?

6. Demostrá que $\dots \in \Omega(\dots)$ es reflexiva y transitiva; y que $f(n) \in \Omega(g(n))$ equivale a $\Omega(g(n)) \subseteq \Omega(f(n))$.

7. Demostrá que $\dots \in \Theta(\dots)$ es de equivalencia; y que $f(n) \in \Theta(g(n))$ equivale a $\Omega(f(n)) = \Omega(g(n))$.

8. Resolvé las siguientes recurrencias:

- (a) $t(n) = \begin{cases} 1 & \text{si } n = 1 \\ 4 & \text{si } n = 2 \\ 8t(n-1) - 15t(n-2) & \text{si } n > 2 \end{cases}$
- (b) $t(n) = \begin{cases} 0 & \text{si } 1 \leq n \leq 2 \\ 1 & \text{si } n = 3 \\ 3t(n-2) - 2t(n-3) & \text{si } n > 3 \end{cases}$
- (c) $t(n) = \begin{cases} 5 & \text{si } n = 1 \\ 3t(n-1) - 2^{n-1} & \text{si } n > 1 \end{cases}$
- (d) $t(n) = \begin{cases} 1 & \text{si } n = 1 \\ t(n-1) + n/2 & \text{si } n > 1 \end{cases}$
- (e) $t(n) = \begin{cases} n & \text{si } 0 \leq n \leq 2 \\ 3t(n-1) - 4t(n-3) & \text{si } n > 2 \end{cases}$
- (f) $t(n) = \begin{cases} 1 & \text{si } 0 \leq n \leq 1 \\ \frac{t(n-1) + t(n-2) + 12n - 16}{2} & \text{si } n \geq 2 \end{cases}$

9. Dada la siguiente recurrencia

$$t(n) = \begin{cases} 4 & \text{si } n = 1 \\ 2t(n/2) + 2n\log_2(n) & \text{si } n > 1 \end{cases}$$

Demostrá que $t(n) \in \mathcal{O}(n\log^2(n))$. Ayuda: primero demostralo para $n = 2^m$.

10. Calculá el orden de complejidad de los siguientes algoritmos:

- (a) **proc** *f1*(**in** $n : \text{nat}$)
 if $n \leq 1$ **then skip**
 else
 for $i := 1$ **to** 8 **do** *f1*($n \text{ div } 2$) **od**
 for $i := 1$ **to** n^3 **do** *operación_de* $\mathcal{O}(1)$ **od**
- (b) **proc** *f2*(**in** $n : \text{nat}$)
 for $i := 1$ **to** n **do**
 for $j := 1$ **to** i **do** *operación_de* $\mathcal{O}(3)$ **od**
 od
 if $n \leq 0$ **then skip**
 else
 for $i := 1$ **to** 4 **do** *f2*($n \text{ div } 2$) **od**
- (c) **proc** *f3*($n : \text{nat}$)
 for $j := 1$ **to** 6 **do**
 if $n \leq 1$ **then skip**
 else
 for $i := 1$ **to** 3 **do** *f3*($n \text{ div } 4$) **od**
 for $i := 1$ **to** n^4 **do** *operación_de* $\mathcal{O}(1)$ **od**
 od

11. Sean K y L constantes, y f el siguiente procedimiento:

```
proc f(in  $n : \text{nat}$ )
  if  $n \leq 1$  then skip
  else
    for  $i := 1$  to  $K$  do f( $n \text{ div } L$ ) od
    for  $i := 1$  to  $n^4$  do operación_de $\mathcal{O}(1)$  od
```

Determiná posibles valores de K y L de manera que el procedimiento tenga orden:

- (a) $\Theta(n^4 \log n)$ (b) $\Theta(n^4)$ (c) $\Theta(n^5)$

12. Calculá el orden de cada uno de los siguientes algoritmos:

- (a) $i := 1;$
 do $i \leq N$
 $c := i;$
 do $c > 1$
 operación_de $\mathcal{O}(1);$
 $c := c/2;$
 od
 $i := i + 1;$
 od
- (b) **do** $n > 0 \wedge m > 0$
 if $n > m$ **then** $n, m := m, n \text{ mod } m$
 else $n, m := n, m \text{ mod } n$
 od
 if $n = 0$ **then return** m
 else return n

13. Escribí algoritmos cuyas complejidades sean (asumiendo que el lenguaje no tiene multiplicaciones ni logaritmos, o sea que no podés escribir **for** $i := 1$ **to** $n^2 + 2 \log n$ **do** ... **od**):

- (a) $n^2 + 2 \log n$ (b) $n^2 \log n$ (c) 3^n