UNIVERSIDAD DE OVIEDO
ESCUELA POLITÉCNICA DE INGENIERÍA de GIJÓN

ALGORITMIA
1ª Evaluación Teórica: Análisis de Algoritmos
6 de octubre de 2021

1. [1.75 Puntos] Sea la tabla adjunta donde n es la talla del problema, $T(n)$ los pasos del algoritmo, $C(n)$ el tiempo en segundos de una implementación en lenguaje C y $P(n)$ el tiempo en segundos de una implementación en lenguaje P. Para todo valor de n se mantiene la razón matemática mostrada en la tabla y se acepta la siguiente equivalencia: "*1 paso = 1 segundo*". Completar, en la zona de respuestas, las frases/expresión con precisión ("*La pregunta es correcta sí y solo sí todas las respuestas son correctas. En caso contrario se considerará totalmente incorrecta*").

n	$T(n)$	$C(n)$	$P(n)$
1	1	0.1	10
3	9	0.9	90
5	25	2.5	250
7	49	4.9	490
9	81	8.1	810
…	…	…	…

Respuesta

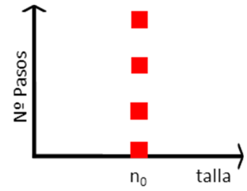
$T(n) \in \boldsymbol{\theta(n^2)}$

$C(n)$ es $\mathbf{0.1}T(n)$

$P(n)$ es $\mathbf{100}C(n)$

Por el **Principio de Invarianza** son "*iguales*".

2. [1.5 Puntos] En la figura adjunta se muestra el número de pasos contabilizados, en *distintas simulaciones*, de un algoritmo para un valor concreto (n_0) de la talla del problema. ¿qué está *diciendo* la figura? Razonar la respuesta.



Respuesta

El algoritmo presenta mejor y peor caso dado que, para un valor concreto de la talla, el número de pasos cambia (no es constante) en función de *algo* que no depende del tamaño del problema, de *algo* que depende de las *instacias*.

3. [1.75 Puntos] Sean las funciones $f(n) = (\sqrt[3]{27})n^3 + 3n^2$ y $g(n) = n^2 \log_2(4) + 2n$, con $n \geq 1$. **Demostrar** la relación de dominancia entre ambas funciones.

Respuesta

$$\lim_{n\to\infty}\left(\frac{f(n)}{g(n)}\right) = \lim_{n\to\infty}\left(\frac{(\sqrt[3]{27})n^3 + 3n^2}{n^2\log_2(4) + 2n}\right) = \lim_{n\to\infty}\left(\frac{3n^2(n+1)}{2n(n+1)}\right) = \lim_{n\to\infty}\left(\frac{3n}{2}\right) = \lim_{n\to\infty}(1.5n) = \infty$$

El límite es $\infty \Longrightarrow f(n)\ domina\ a\ g(n)$.

Planteado al revés

$$\lim_{n\to\infty}\left(\frac{g(n)}{f(n)}\right) = \lim_{n\to\infty}\left(\frac{n^2\log_2(4) + 2n}{(\sqrt[3]{27})n^3 + 3n^2}\right) = \lim_{n\to\infty}\left(\frac{2n(n+1)}{3n^2(n+1)}\right) = \lim_{n\to\infty}\left(\frac{2}{3n}\right) = \lim_{n\to\infty}\left(\frac{1.5}{n}\right) = 0$$

El límite es $0 \Longrightarrow g(n)\ es\ domina\ por\ f(n)$

4. [5 Puntos] Sea el algoritmo de la figura adjunta. Calcular su complejidad, siendo riguroso en la aplicación y explicación de los pasos y cálculos seguidos. **Resolver en una hoja aparte**.

```
funcion Examen(V[1...m]: vector de enteros; m: entero) retorna entero
    var suma, i: entero fvar
    si (m ≥ 1) entonces
        si (par(m)) entonces
            retorna m + Examen(V, m – 2);
        sino
            suma = 0; i = m;
            mientras (1 ≤ i) hacer
                suma = suma + i;
                i = i – 2;
            fmientras
            retorna suma;
        fsi
    sino
        retorna m;
    fsi
ffuncion
```

Solución

La talla del problema es m: número de elementos del vector *V* que son tratados (su dimensión). El algoritmo no presenta mejor/peor caso: fijada cualquier talla del problema el algoritmo siempre hace el mismo número de pasos, independientemente del contenido (instancias) del vector. Pero sí que, en función de que la talla sea par o $impar$, hace unos pasos u otros. En consecuencia, se debe calcular la más grande/pequeña de la acotación inferior/superior. Se procede entonces a:

- Acotar para el supuesto "m es par", que denotamos por $T_{par}(m)$
  Es un fragmento de código recursivo que mantiene la paridad de m inalterada durante las sucesivas llamadas. Su ecuación es:

$$T_{par}(m) = \begin{cases} c_1 & si\ (1 > m) \\ T_{par}(m-2) + c_2 & si\ (1 \leq m) \end{cases}$$

  En la solución del ejercicio 14 del "*boletín de problemas resueltos*" está el desarrollo correspondiente a la ecuación anterior, resultado que $T_{par}(m) \in \theta(m)$.

- Acotar para el supuesto "m es impar", que denotamos por $T_{impar}(m)$
  Es un fragmento de código iterativo, cuyo número de pasos se puede aproximar por

$$T_{impar}(m) \cong 3 + \sum_{i=0}^{\frac{m}{2}} 2 = 3 + 2\left(\frac{m}{2}+1\right) = 3 + m + 1 = 4 + m \in \theta(m)$$

  donde el 3 corresponde a las operaciones básicas/elementales "suma = 0", "i = m" y "retorna suma", que se hacen una sola vez, y el 2 se refiere a los pasos "suma = suma + i" e "i = i – 2", cuya ejecución se repite $\left(\frac{m}{2}+1\right)$ veces. La comparación "$1 \leq i$" no se ha contabilizado al considerarla ligada al "iterador". Si se hubiese hecho, igual de correcto, el número 2 pasaría a ser un 3.
  En este supuesto m es impar y $\frac{m}{2}$ denota la división entera (m div 2, como en el resto de ejercicios presentados/resueltos en clase). Consecuentemente, $2\left(\frac{m}{2}+1\right) = 2\left(\frac{m+1}{2}\right) = m + 1$

Al coincidir las cotas se puede concluir que $T(m) \in \theta(m)$.