Análisis de Algoritmos

Luis Manuel Román García & María Fernanda Mora Alba

2 de septiembre de 2015

1. Probar si las siguientes funciones pertenecen a $\Theta(n^2)$

Definición Decimos que $f \in \Theta(g(n))$ si y sólo si $\exists \quad c_1, c_2 \in \mathbb{R}$ y $n_0 \in \mathbb{N}$ tal que $\forall n \geq n_0$ se cumple que $c_2 g(n) \leq f(n) \leq c_1 g(n)$

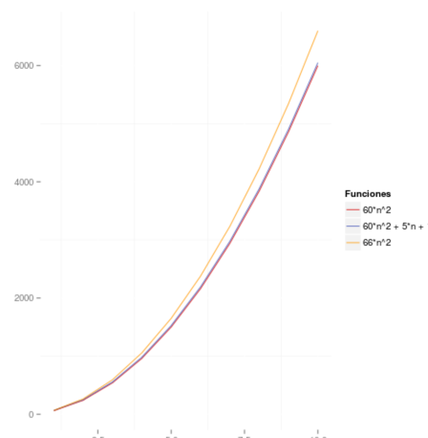
1.1. $60n^2 + 5n + 1$

Solución

$$60n^2 + 5n + 1 \leq c_1 n^2 \iff 60 + \frac{5}{n} + \frac{1}{n^2} \leq c_1 \quad (1) \qquad 60n^2 + 5n + 1 \geq c_2 n^2 \iff 60 + \frac{5}{n} + \frac{1}{n^2} \geq c_2 \quad (2)$$

De (1) es fácil ver que $\forall n \geq 1$ se tiene que $60 + \frac{5}{n} + \frac{1}{n^2} \leq 60 + 5 + 1 = 66$ y de (2) tenemos que $\forall n \geq 1$ se cumple que $60 + \frac{5}{n} + \frac{1}{n^2} \geq 60$. De aquí que tomando $n_0 = 1, c_1 = 66, c_2 = 60$ se cumple que $\forall n \geq n_0 \Rightarrow 60n^2 \leq 60n^2 + 5n + 1 \leq 66n^2$

$\therefore \quad 60n^2 + 5n + 1 \in \Theta(n^2)$ $\square$

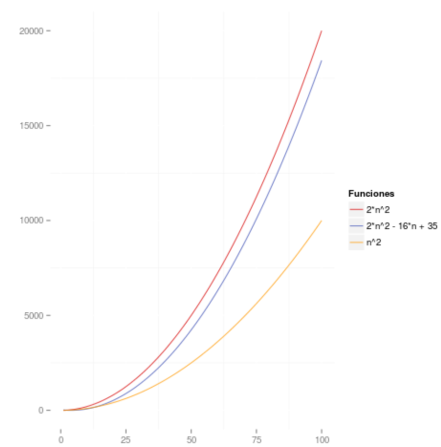


1.2. $2n^2 - 16n + 35$

Solución

$$\begin{array}{ccc} 2n^2 - 16n + 35 \leq c_1 n^2 & & 2n^2 - 16n + 35 \geq c_2 n^2 \\ \Leftrightarrow & (3) \qquad\qquad & \Leftrightarrow & (4) \\ 2 - \dfrac{16}{n} + \dfrac{35}{n^2} \leq c_1 & & 2 - \dfrac{16}{n} + \dfrac{35}{n^2} \geq c_2 \end{array}$$

En (3), para $n \geq 4$ se cumple que $\frac{16}{n} \geq \frac{35}{n^2}$ y por ende $2 \geq 2 - (\frac{16}{n} - \frac{35}{n^2})$. Por otro lado en la desigualdad (4) para valores de $n \geq 16$ se tiene que $2 - \frac{16}{n} + \frac{35}{n^2} \geq 2 - 1 + \frac{35}{n^2} \geq 2 - 1 = 1$. De aquí que si tomamos $n_0 = 16, c_1 = 2, c_2 = 1$ tenemos que $\forall n \geq n_0 \Rightarrow n^2 \leq 2n^2 - 16n + 35 \leq 2n^2 \therefore \quad 2n^2 - 16n + 35 \in \Theta(n^2)$ $\square$



1.3. $3n^2 + 2nlog(n)$

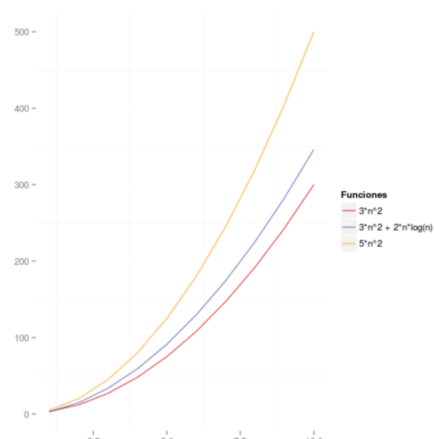
Solución

Notemos que la función $f(n) = n - log(n)$ es creciente para $n \geq 1$ [1] esto quiere decir que $\forall n \geq 1; \quad n \geq log(n)$ luego entonces se sigue que:

$$\begin{aligned} 3n^2 + 2nlog(n) &\leq 3n^2 + 2n * n \\ &= 3n^2 + 2n^2 \\ &= 5n^2 \end{aligned} \tag{5}$$

$$3n^2 + 2nlog(n) \geq 3n^2 \tag{6}$$

Ambas desigualdades se cumplen $\forall n \geq 1$[2]. De esta manera si elegimos $n_0 = 1, c_1 = 5, c_2 = 3$ tenemos que $\forall n \geq n_0 \Rightarrow 3n^2 \leq 3n^2 + 2nlog(n) \leq 5n^2 \therefore$ $3n^2 + 2nlog(n) \in \Theta(n^2)$ $\square$



[1] Es fácil probar esto ya que $\frac{df}{dn}f(n) = 1 - \frac{1}{n} \geq 0 \quad \forall n \geq 1$

[2] (6) se sigue de que $nlog(n) \geq 0 \quad \forall n \geq 1$

1.4. $\sum_{i=1}^{n} 2i$

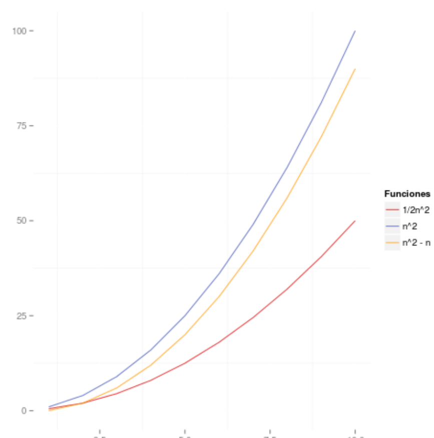
Solución

Es fácil ver que $\sum_{i=1}^{n} 2i = 2\sum_{i=1}^{n} i = 2\frac{n(n-1)}{2} = n(n-1)$ de aquí se sigue que

$$\begin{aligned} n(n-1) &= n^2 - n \\ &\leq n^2 \end{aligned} \tag{7}$$

$$n(n-1) = n^2 - n \tag{8}$$

Para acotar (8), simplemente notamos que $\forall n \geq 2$ se tiene que $n^2 - n \geq \frac{n^2}{2}$ [3] de aquí tomamos $n_0 = 2, c_1 = 1, c_2 = \frac{1}{2}$ y tenemos que $\forall n \geq n_0 \Rightarrow \frac{1}{2}n^2 \leq n^2 - n \leq n^2 \therefore \quad n^2 - n \in \Theta(n^2)$ $\square$



[3]Es fácil probar esto ya que $\frac{df}{dn}(n^2 - n - \frac{n^2}{2}) = \frac{df}{dn}(\frac{n^2}{2} - n) = n - 1 \geq 0 \quad \forall n \geq 1$ lo que implica que $n^2 - n - \frac{n^2}{2}$ es una función creciente para $n \geq 1$

1.5.

for i = *1 to n* **do**
  for j = *1 to i* **do**
    for k = *1 to j* **do**
      x = x + 1;
    end
  end
end

Este algoritmo se puede rescribir como una triple suma $\sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{j} k$. Y es claro que $\sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{j} k \leq n\sum_{j=1}^{n}\sum_{k=1}^{j} k \leq n^2 \sum_{k=1}^{j} k \leq n^3$. Ahora bien, para encontrar la cota inferior hay que notar que cada ciclo implica un número n de operaciones, de aquí que el algoritmo también esté acotado inferiormente por n^3 Lo que implica que no pertence a $\Theta(n^2)$

2. Ordenar las siguientes funciones con base a crecimiento asintótico

Este ejercicio decidimos abordarlo primero de manera **teórica** porque nos pareció muy útil para ejercitar las notaciones de orden vistas en clase y entender cómo se usan para comparar el crecimiento de dos funciones cualquiera. Asimismo nos permitió ver que incluso pueden ser usadas para construir clases de equivalencia de funciones considerando la relación matemática que puede ser definida con el *crecimiento asintótico.*

Además, nos dimos cuenta que para abordar un enfoque gráfico debíamos primero agrupar las funciones por magnitudes de crecimiento, pues al tratar de graficarlas todas en el mismo plano muchas de estas se perdían visualmente.

Posteriormente lo abordaremos de manera gráfica.

2.1. Enfoque teórico

Consideremos las siguientes observaciones:

1. *Grosso modo* las funciones exponenciales crecen más rápido que los polinomios, estos más rápido que las funciones lineales, las líneales más rápido que las logarítmicas, que a su vez crecen más rápido que las funciones constantes. Estas afirmaciones pueden probarse simplemente tomando el límite del cociente de las funciones a comparar:

$$\lim_{n\to\infty} \frac{g_1}{g_2} = +\infty$$

$$\lim_{n\to\infty} \frac{g_1}{g_2} = 0$$

. En el primer caso decimos que g_1 crece asintóticamente más rápido que g_2 y en el segundo que g_2 crece asintóticamente más rápido que g_1.

2. La base del logaritmo es desdeñable en el largo plazo (cuando $n \to \infty$) pero el grado del polinomio y la base de la potencia no lo son.

3. Tomando en cuenta las observaciones anteriores hicimos una pre-clasificación de las funciones por familas: *constantes, logarítmicas, polinómicas, exponenciales, factoriales.*

4. Posteriormente, dentro de cada clase, ordenamos nuevamente a los integrantes según su crecimiento asintótico. Esto se hizo usando varias técnicas: ya sea usando el criterio del límite expuesto en 1. (ayudándonos de la Regla del Hopital para calcular dicho límite), o bien, usando directamente la notación Ω i.e. $f(n) \in \Omega(g(n)) \iff \exists c > 0, n_0 \in \mathbb{N} : 0 \leq cg(n) \leq f(n), \forall n \geq n_0$, o bien la notación Θ, i.e. $f(n) \in \Theta(g(n)) \iff \exists c_1, c_2 > 0, n_0 \in \mathbb{N} : c_1 g(n) \leq f(n) \leq c_2 g(n), \forall n \geq n_0$.

5. Para el caso especial de las exponenciales, se usó el siguiente resultado: Si s_n, t_n son sucesiones que cumplen que $s_n \leq t_n$ entonces $c^{s_n} \leq c^{t_n}, \forall c > 0$. Esto fue útil porque si dos funciones la misma base entonces para comparlas sólo basta comparar los exponentes. En el caso de este ejercicio los exponentes resultaban ser funciones que ya sabíamos cómo comparar: *lineales, logarítmicas, etc.*

6. En el caso de los factoriales nos ayudó la fórmula de Stirling así como notar lo siguiente:

$$log(n!) = log(n(n-1)(n-2)... * 1) = log(n) + log(n-1) + ... + log(1) \leq nlog(n)$$

por ser $log(n)$ una función no decreciente.

7. Notar que $log * x$ denota el logartimo iterado de x. Intuitivamente denota el número de iteraciones necesarias para que el logaritmo sea a a lo más 1. Entonces es una función escalonada cuya imagen está contenida en los naturales. Sin embargo al graficar esta función en base 2 nos dimos cuenta que crece muy lento, mucho más lento que $log(x)$. De hecho vimos que para n relevante en la práctica esta función no sobrepasa el valor de 5. De modo que la definición del logaritmo iterado es la siguiente:

$$log * n = log * (log(n)) + 1 \ \forall n > 1$$

$$log * n = 0 \ \forall n \leq 1$$

8. Para llevar a cabo estas comparaciones nos fueron útiles las siguientes identidades:

$$n^2 = 4^{log(n)}$$

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

$$1 = n^{\frac{1}{log(n)}}$$

$$n^{log(log(n)} = log(n)^{log(n)}$$

De modo que el orden de las funciones dadas queda así, en donde cada $>$ representa una clase de equivalencia:

$$2^{2^{n+1}} > 2^{2^n} > (n+1)! > n! > e^n > n2^n$$

$$>$$

$$2^n > (3/2)^n > n^{log(log(n))} = log(n)^{log(n)} > (log(n))! > n^3$$

$$>$$

$$n^2 = 4^{log(n)} > nlog(n) = log(n!) > n = 2^{log(n)}$$

$$>$$

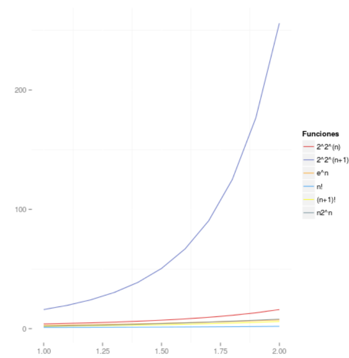
$$\sqrt{2}^{logn} > 2^{\sqrt{2log(n)}} > (log(n))^2 > ln(n) > \sqrt{log(n)} > ln(ln(n))$$

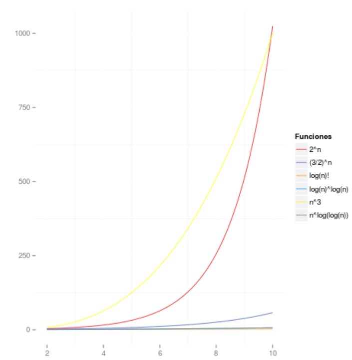
$$>$$

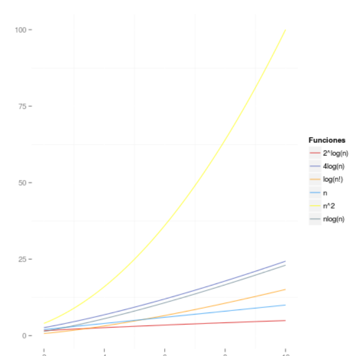
$$2^{log*n} > log*(log(n)) = log*n > log(log*n) > 1 = n^{1/log(n)}$$

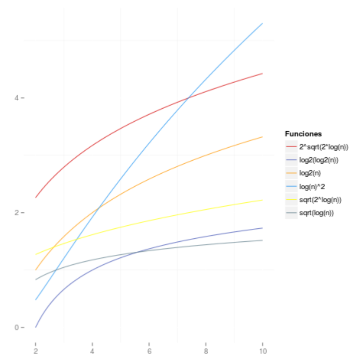
2.2. Enfoque gráfico

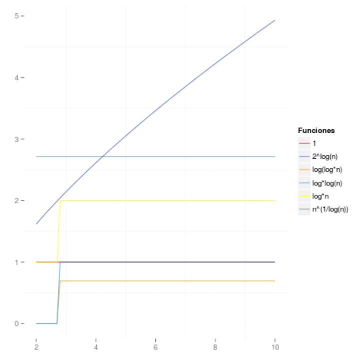
Las funciónes dentro de cada conjunto crecen todas más rápido que todas las del conjunto siguiente.








Funciones
2^sqrt(2*log(n))
log2(log2(n))
log2(n)
log(n)^2
sqrt(2^log(n))
sqrt(log(n))


Funciones
1
2^log(n)
log(log*n)
log*log(n)
log*n
n^(1/log(n))