Notación Asintótica

Programación Avanzada

Orden de crecimiento asintótico

- Ayuda a:
  - Identificar el comportamiento de un algoritmo en el peor, el mejor o el caso promedio.
  - Tener una idea del comportamiento del algoritmo cuando se escala la entrada.
  - Identificar clases de algoritmos que tengan comportamientos comparables.
  - Expresiones como $1.62n^2 + 3.5n + 8$ no tienen generalmente mucho significado, ya que dependen de parámetros de representación intermedia y del hardware utilizado.

Notación asintótica

- Sirve para tener una primera aproximación del desempeño que describe un algoritmo ante instancias grandes de los problemas.

- Razonablemente independiente de:

  - detalles del hardware,
  - lenguajes de programación, compilador, etc.

- "O grande" fue utilizada por primera vez por Bachmann (1892) y popularizada por Landau, por lo que en ocasiones se le conoce también como "notación de Landau".

- Knuth describió las notaciones Θ y Ω para completar la notación O.

- Se utilizan en general funciones cuyos dominios están en el conjunto de los naturales.

Notación asintótica

- Comportamiento asintótico: comportamiento cuando $n \to \infty$, donde n es el tamaño del problema.
- Tres notaciones básicas:

$f \sim g$ ("f y g son asintóticamente equivalentes")

$f \preceq g$ ("f está asintóticamente dominada por g ")

$f \asymp g$ ("f y g están respectivamente asintóticamente acotadas")

Notación asintótica

$$\boxed{f \sim g \quad \text{significa} \quad \lim_{n\to\infty} \frac{f(n)}{g(n)} = 1}$$

Ejemplo: $3x^2 + 2x + 1 \sim 3x^2$

$\sim$ es una relación de equivalencia:

- Transitiva: $(x \sim y) \wedge (y \sim z) \Rightarrow (x \sim z)$
- Reflexiva: $x \sim x$
- Simétrica: $(x \sim y) \Rightarrow (y \sim x)$

Idea básica: tomar en cuenta solo los términos principales, ignorando aquellos que crecen más lento.

Notación asintótica

$$f \preceq g \quad \text{significa} \quad \limsup_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$

i.e., $\dfrac{f(n)}{g(n)}$ está eventualmente acotado por un valor finito.

Idea básica: *f* crece más lento que *g*, o al menos tan rápido como *g*.

$\preceq$ es un pre-orden (o quasi-orden):

Transitivo: $(f \preceq g) \wedge (g \preceq h) \Rightarrow (f \preceq h)$

Reflexivo: $f \preceq f$

Notación asintótica

Escribir $f \asymp g$ cuando hay constantes positivas c_1, c_2 tales que,

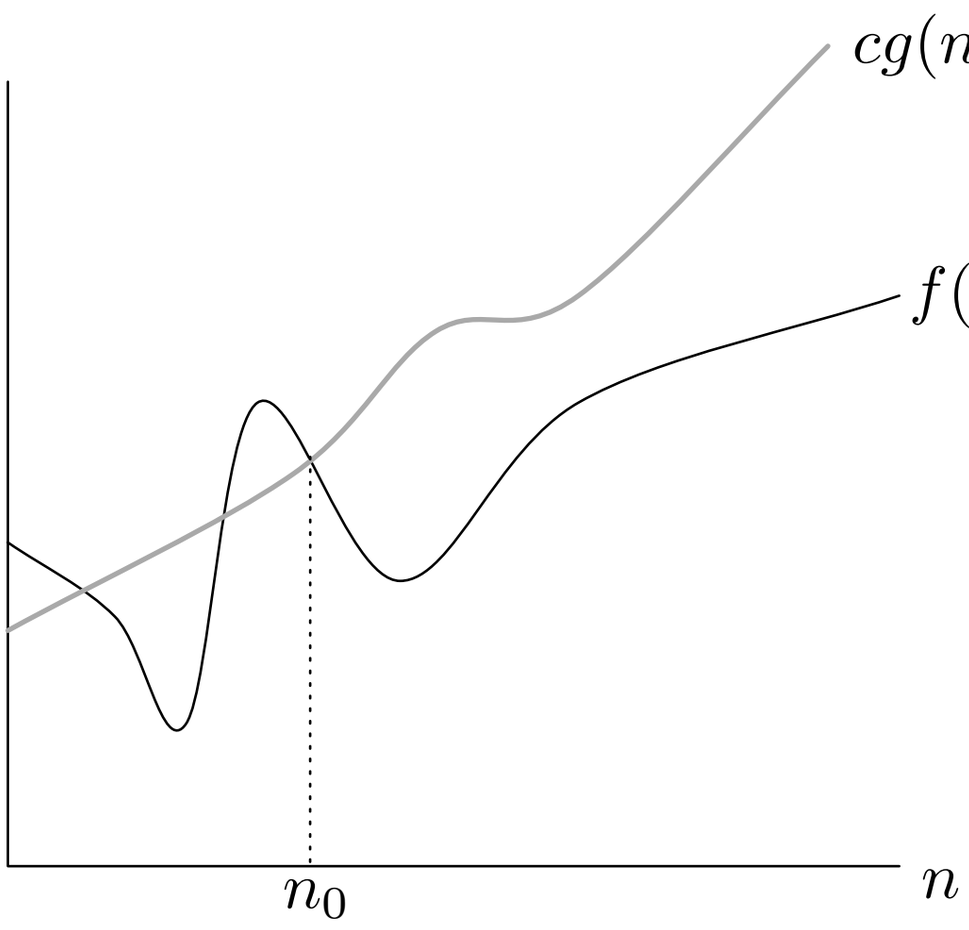
$$c_1 \leq \frac{f(n)}{g(n)} \leq c_2$$

para una n suficientemente grande.

- Ejemplos:
  - $n \asymp 2n$
  - $n \asymp (2 + sin\pi n)n$
- $\asymp$ es una relación de equivalencia.

Notación O grande

$$T(n) = O(f(n)) \quad \text{ssi} \quad \exists (c, n_0) \in \mathbb{R}^{+*} \times \mathbb{N} \mid \forall\, n \geq n_0 \quad T(n) \leq cf(n)$$



$$f(n) = O(g(n))$$

- Se lee: "T(n) es de orden f(n)"
- T(n) no crece más rápido que f(n)
- Correctamente diríamos que T(n) ∈ O(f(n)).

Notación O grande

Convención:

Un conjunto en una formula representa una función anónima en el conjunto.

Ejemplo:

$$f(n) = n^3 + O(n^2)$$

significa,

$$f(n) = n^3 + h(n)$$

para alguna $h(n) \in O(n^2)$.

Ejemplo:

$$n^2 + O(n) = O(n^2)$$

significa que para cualquier $f(n) \in O(n)$:

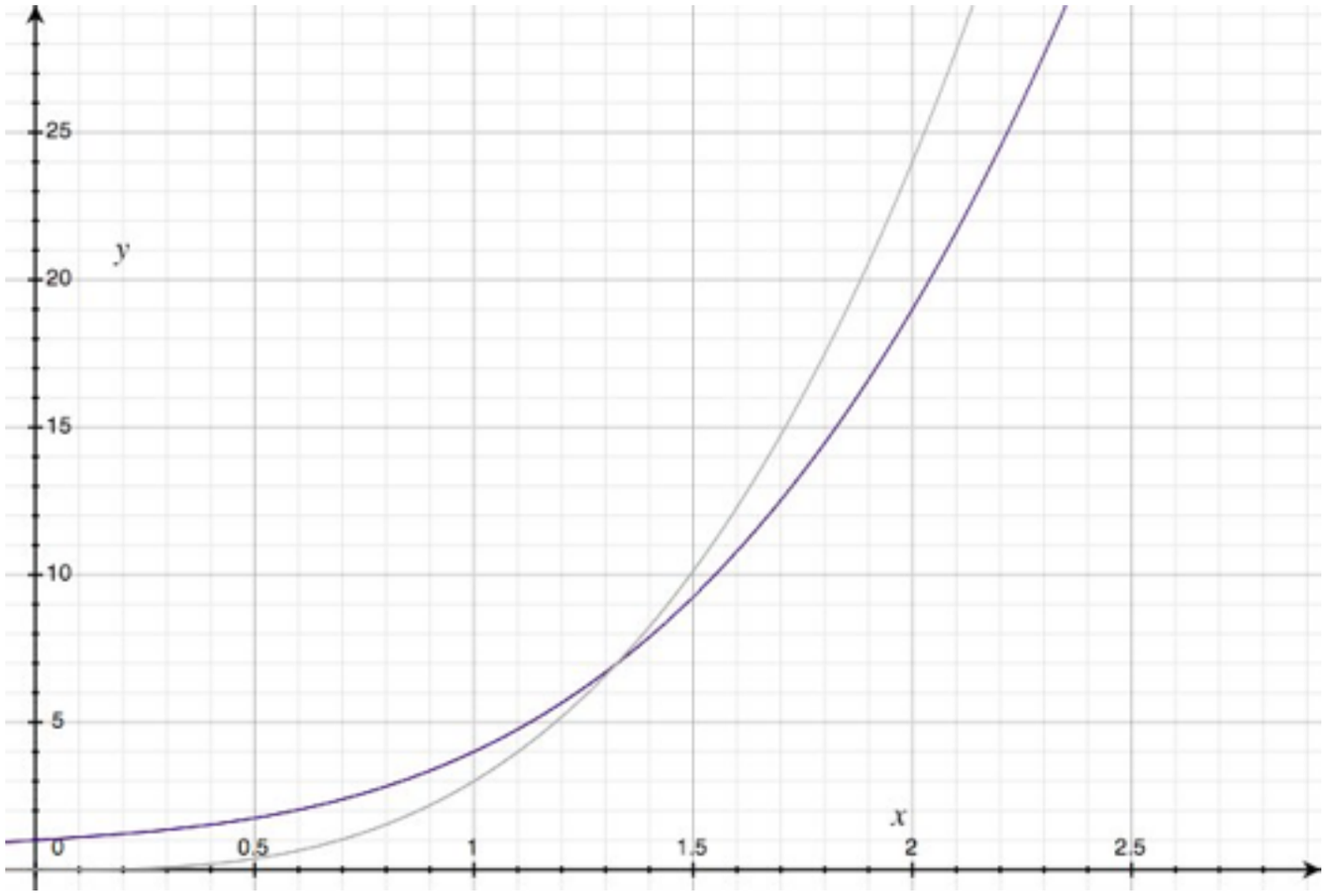
$$n^2 + f(n) = h(n)$$

para alguna $h(n) \in O(n^2)$.

Notación O grande

$$T(n) = 2n^3 + n + 1 = O(n^3)$$

porque haciendo *$(c, n_0) = (3, 2)$* verificamos que para *$n \geq 2$*



$$T(n) = 2n^3 + n + 1$$

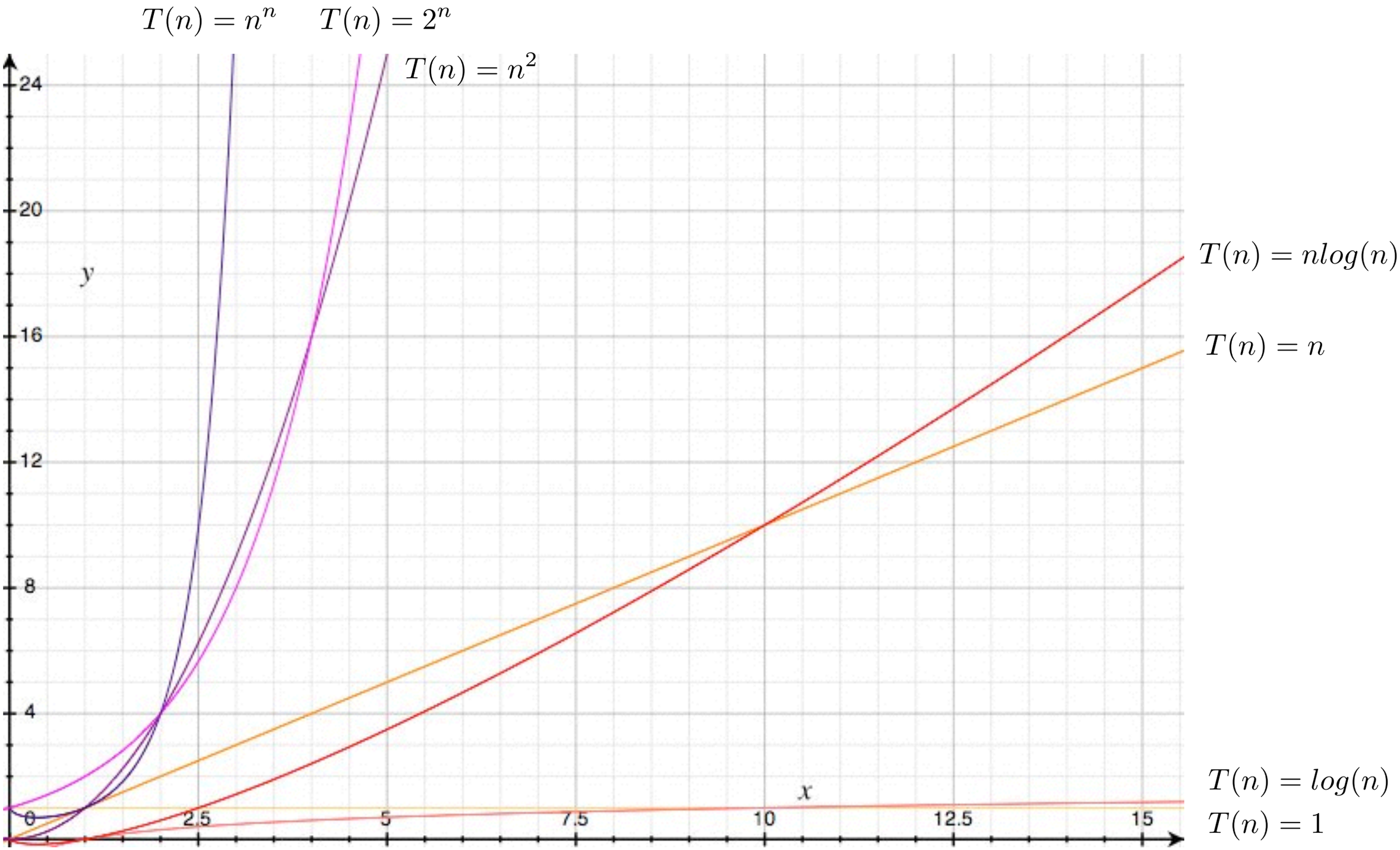
$$\leq 3n^3$$

Notación O grande

- La notación O grande permite definir un límite superior al crecimiento de una función.

- Establece un "**orden**" entre funciones:

$$1 \leq \log n \leq n \leq n \log n \leq n^2 \leq 2^n \leq n^n$$


$T(n) = n^n$
$T(n) = 2^n$
$T(n) = n^2$
$T(n) = nlog(n)$
$T(n) = n$
$T(n) = log(n)$
$T(n) = 1$
y
x
24
20
16
12
8
4
0
2.5
5
7.5
10
12.5
15

Notación O grande

- Considerar funciones mayorantes que sean lo más pequeñas posibles:

$$\begin{aligned} T(n) &= 2n^3 + n + 1 \\ T(n) &= O(n^4) \\ T(n) &= O(2^n) \\ T(n) &= O(n!) \end{aligned}$$

- Si T(n) es un polinomio de grado g, entonces $T(n) = O(n^g)$: se pueden "olvidar" los términos de grado inferior y las constantes.

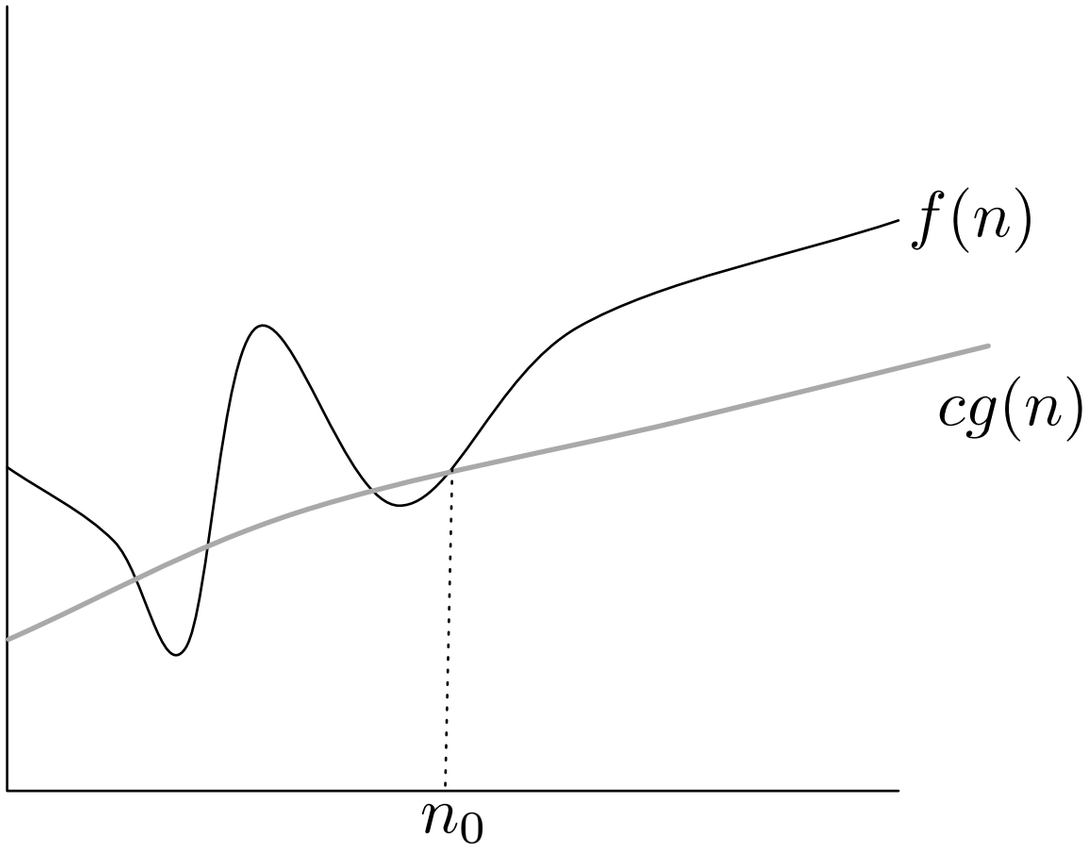
- Usar la clase de funciones más chica: 2n es O(n), aunque sea también $O(n^2)$

- Usar la expresión más simple

Notación Ω

- Presenta la cota asintótica inferior de una función.

$$T(n) = \Omega(f(n)) \quad \text{ssi} \quad \exists (c, n_0) \in \mathbb{R}^{+*} \times \mathbb{N} \mid \forall\, n \geq n_0 \quad T(n) \geq cf(n)$$



$$f(n) = \Omega(g(n))$$

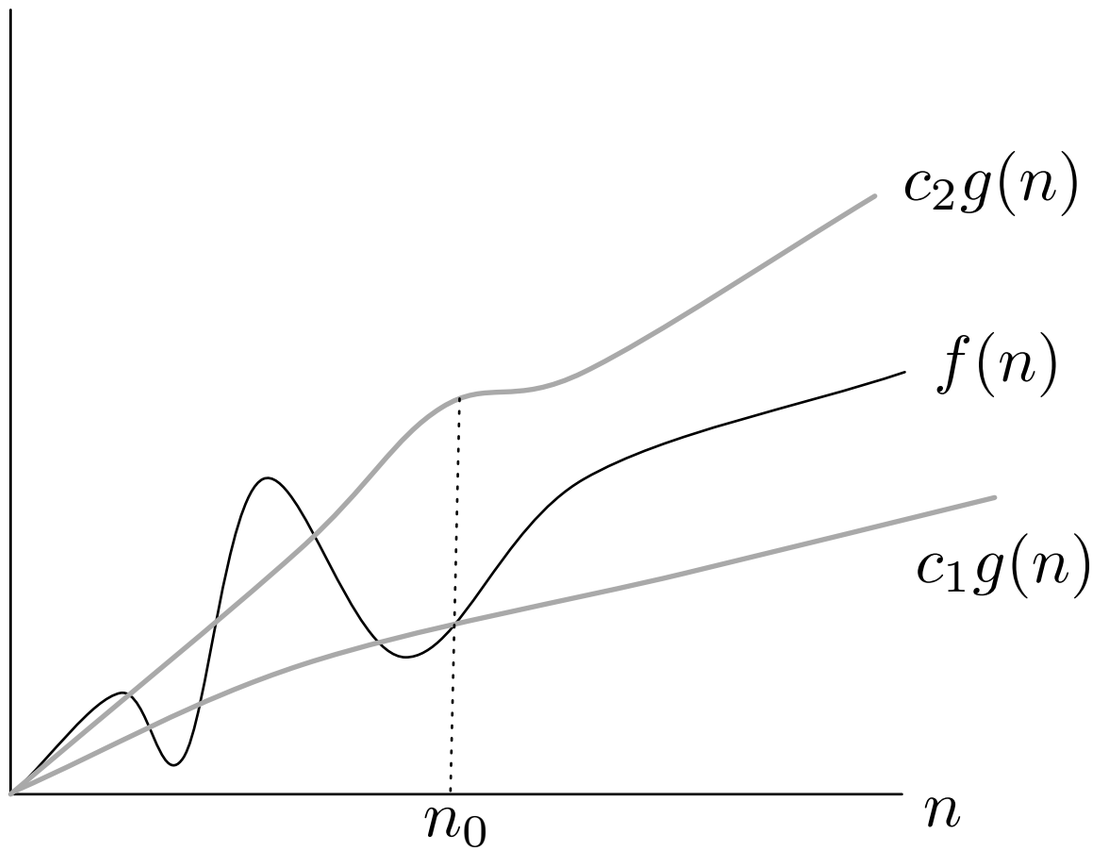
- Se usa para acotar el tiempo de ejecución en el mejor caso.

- Si un algoritmo es Ω(g(n)), decimos que sin importar el tamaño n de la entrada, el tiempo de cálculo para esa entrada es al menos una constante multiplicativa de g(n), para una n suficientemente grande.

Notación Θ

$$T(n) = \Theta(f(n)) \quad \text{ssi} \quad \exists (c_1, c_2, n_0) \in \mathbb{R}^{+*} \times \mathbb{R}^{+*} \times \mathbb{N}$$

$$\text{tal que} \quad \forall\ n \geq n_0 \quad 0 \leq c_1 f(n) \leq T(n) \leq c_2 f(n)$$



$$f(n) = \Theta(g(n))$$

Notación Θ

$$\frac{1}{2}n^2 - 3n = \Theta(n^2)$$

Determinar las constantes positivas c_1, c_2, y n_0 tales que

$$c_1 n^2 \leq \frac{1}{2}n^2 - 3n \leq c_2 n^2 \quad \forall n \geq n_0$$

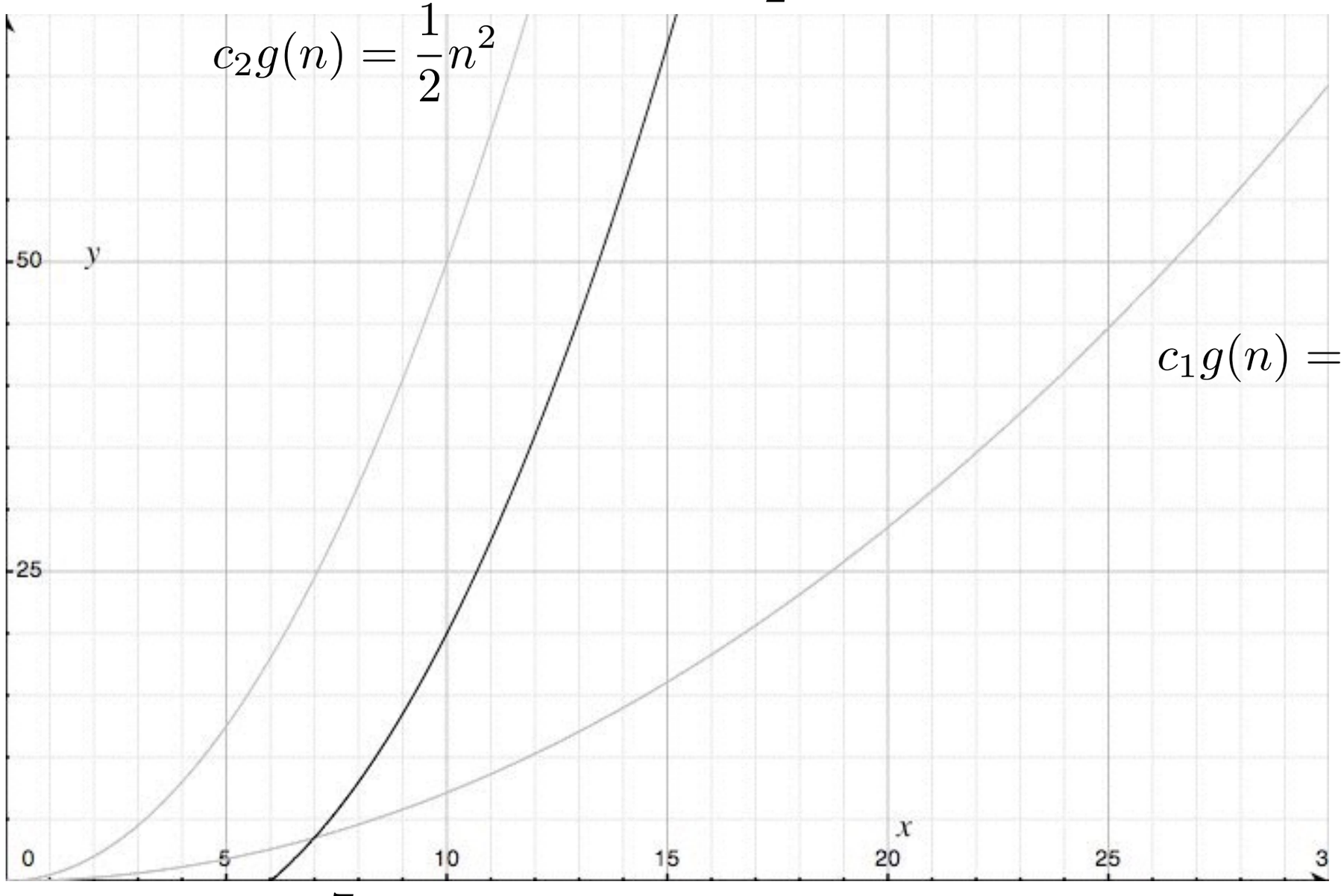
Dividiendo por n^2 tenemos:

$$c_1 \leq \frac{1}{2} - \frac{3}{n} \leq c_2$$

$$\forall n \geq 1, \frac{1}{2} - \frac{3}{n} \leq \frac{1}{2} \qquad \forall n \geq 7, \frac{1}{2} - \frac{3}{n} \geq \frac{1}{14}$$

Notación Θ

$$f(n) = \frac{1}{2}n^2 - 3n$$



$n_0 = 7$

$$f(n) = \Theta(n^2)$$

Propiedades de la notación asintótica

- Transitividad

$$\text{Si } f(n) = O(g(n)) \text{ y } g(n) = O(h(n)), \text{ entonces } f(n) = O(h(n))$$

$$\text{Si } f(n) = \Omega(g(n)) \text{ y } g(n) = \Omega(h(n)), \text{ entonces } f(n) = \Omega(h(n))$$

$$\text{Si } f(n) = \Theta(g(n)) \text{ y } g(n) = \Theta(h(n)), \text{ entonces } f(n) = \Theta(h(n))$$

- Adición

$$\text{Si} f(n) = O(h(n)) \text{ y } g(n) = O(h'(n)), \text{ entonces}$$

$$f(n) + g(n) = O(\max(h(n), h'(n)))$$

- Multiplicación

$$\text{Si } f(n) = O(h(n)) \text{ y } g(n) = O(h'(n)), \text{ entonces}$$

$$f(n)g(n) = O(h(n)h'(n))$$

Notaciones estándar y funciones comunes

- Funciones floor y ceiling (redondeo):
  - $\lfloor x \rfloor$: mayor entero menor que o igual a x.
  - $\lceil x \rceil$: menor entero mayor que o igual a x.

- Para todo real x: $x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1.$
- Para todo entero n: $\lceil n/2 \rceil + \lfloor n/2 \rfloor = n$
- Para todo real $n \geq 0$ y enteros $a,b > 0$,

$$\lceil \lceil n/a \rceil / b \rceil = \lceil n/ab \rceil, \qquad \lceil a/b \rceil \leq (a + (b-1))/b,$$

$$\lfloor \lfloor n/a \rfloor / b \rfloor = \lfloor n/ab \rfloor, \qquad \lfloor a/b \rfloor \geq (a + (b-1))/b,$$

Notaciones estándar y funciones comunes

- Funciones de aritmética modular:

  Para cualquier entero *a* y cualquier entero positivo *n*, el valor *a mod n* es el **residuo** del cociente *a/n* :

$$a \ mod \ n = a - \lfloor a/n \rfloor n.$$

Notaciones estándar y funciones comunes

- Polinomios:

  Dado un entero positivo *d*, un polinomio en *n* de grado *d* es una función *p(n)* de la forma:

$$p(n) = \sum_{i=0}^{d} a_i n^i$$

  donde las constantes $a_0, a_1, \ldots, a_d$ son los coeficientes del polinomio y $a_d \neq 0$.

- Un polinomio es asintóticamente positivo ssi $a_d > 0$.
- Para un polinomio asintóticamente positivo $p(n)$ de grado d, $p(n) = \Theta(n^d)$.
- Decimos que una función $f(n)$ está acotada polinomialmente si $f(n) = O(n^k)$ para alguna constante k.

Notaciones estándar y funciones comunes

- Exponenciales:

  Para todo *a>0*, *m*, y *n*, tenemos las identidades siguientes:

$$\begin{aligned} a^0 &= 1, \\ a^1 &= a, \\ a^{-1} &= 1/a, \\ (a^m)^n &= a^{mn}, \\ (a^m)^n &= (a^n)^m, \\ a^m a^n &= a^{m+n}. \end{aligned}$$

- Para toda *n* y *a≥ 1*, la función a^n es monotónicamente creciente en *n*.
- Cuando convenga supondremos $0^0=1$.
- Las tasas de crecimiento de polinomios y exponenciales se pueden relacionar del hecho que para toda constante real *a* y *b* tal que *a>1*,

  $\lim_{n\to\infty} \frac{n^b}{a^n} = 0$ se concluye que $n^b = o(a^n)$ que indica que cualquier función con base estrictamente mayor a 1, crece mas rapido que cualquier polinomial.

Notaciones estándar y funciones comunes

- Funciones floor y ceiling.
- Funciones de aritmética modular (operación módulo)
- Polinomios.
- Exponenciales.
- Logaritmos.
- Factoriales.
- ...

Notaciones estándar y funciones comunes

- Con un procesador que ejecuta un millón de instrucciones de alto nivel por segundo:

	n	$n\log_2 n$	n^2	n^3	1.5^n	2^n	n!
n=10	< 1s	< 1s	<1s	<1s	<1s	<1s	4s
n=30	< 1s	< 1s	<1s	<1s	<1s	18 min	10^{25} años
n=50	< 1s	<1s	<1s	<1s	11 min	36 años	--
n=100	< 1s	<1s	<1s	1 s	12,892 años	10^{17} años	--
n=1000	< 1s	< 1s	1 s	18 min	--	--	--
n=10000	< 1s	< 1s	2 min	12 días	--	--	--
n=100000	< 1s	2 s	3 horas	32 años	--	--	--
n=1000000	1 s	20 s	12 días	31,710 años	--	--	--