Analisis y Diseño de Algoritmos

Juan Gutiérrez

August 19, 2022

Insertion sort

El Insertion sort

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 . . j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

Figure 1: Tomada del libro Cormen, Introduction to Algorithms

El Insertion sort

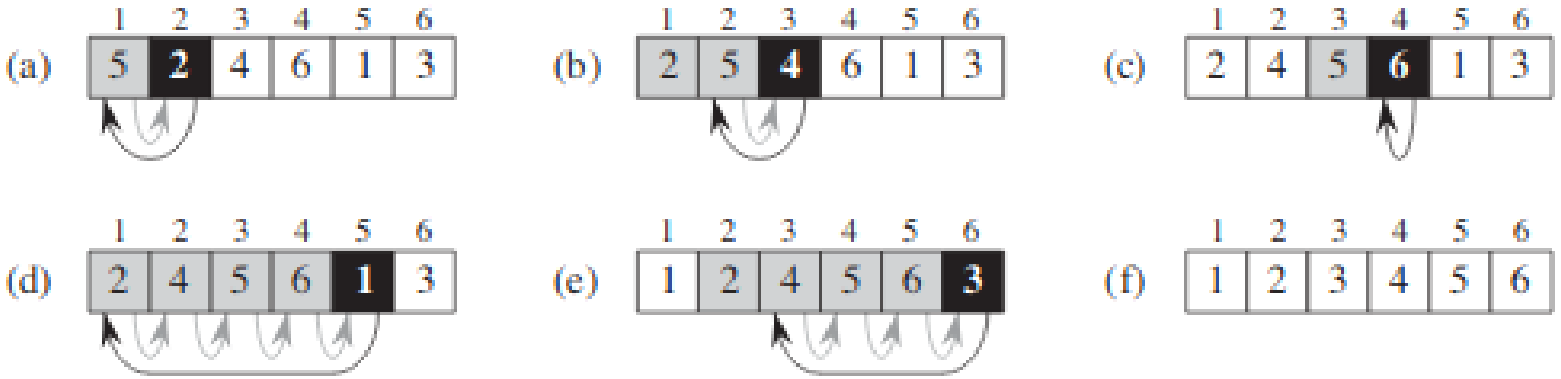


Figure 2: Tomada del libro Cormen, Introduction to Algorithms

El Insertion sort

Invariante: Al inicio de cada iteracion del for de las lineas 1–8, el subarreglo $A[1..j-1]$ consiste en los elementos de $A[1..j-1]$ pero ordenados de manera no descendente.

El Insertion sort

INSERTION-SORT(A)	*cost*	*times*
1 **for** $j = 2$ **to** $A.length$	c_1	n
2 $key = A[j]$	c_2	$n-1$
3 // Insert $A[j]$ into the sorted sequence $A[1..j-1]$.	0	$n-1$
4 $i = j-1$	c_4	$n-1$
5 **while** $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
6 $A[i+1] = A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
7 $i = i-1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
8 $A[i+1] = key$	c_8	$n-1$

Figure 3: Tomada del libro Cormen, Introduction to Algorithms

Notacion O-grande

$$O(g(n)) = \{f(n) : \text{existen constantes positivas } c, n_0$$
$$\text{tales que } 0 \leq f(n) \leq cg(n) \text{ para todo } n \geq n_0\}$$

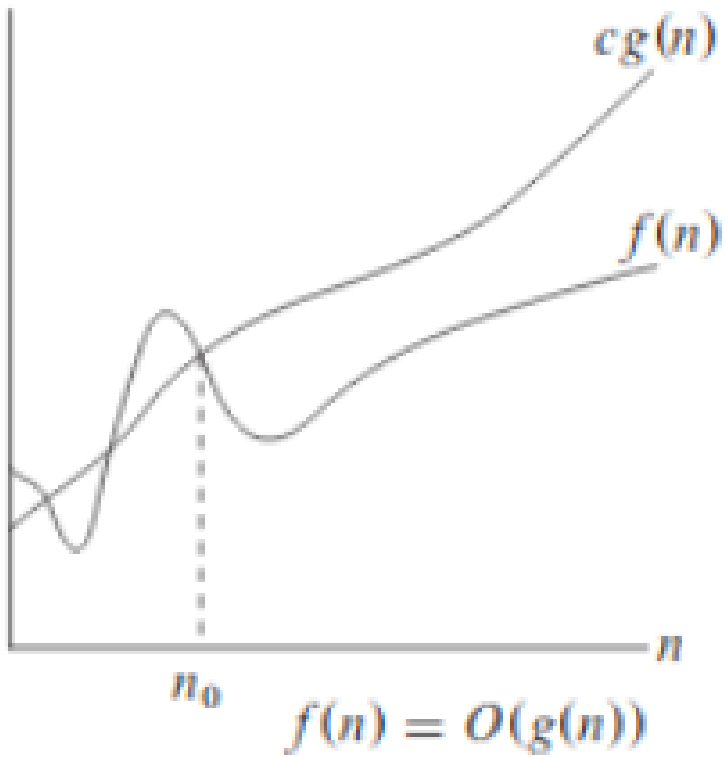


Figure 4: Tomada del libro Cormen, Introduction to Algorithms

Notacion O-grande

Ejemplo 2.1. *Probar que* $n^2 + 10n + 2 = O(n^2)$

Notacion O-grande

Ejemplo 2.2. *Probar* $n^2/2 + 3n = O(n^2)$

Notacion O-grande

Ejemplo 2.3. *Probar* $n/100$ *no es* $O(1)$.

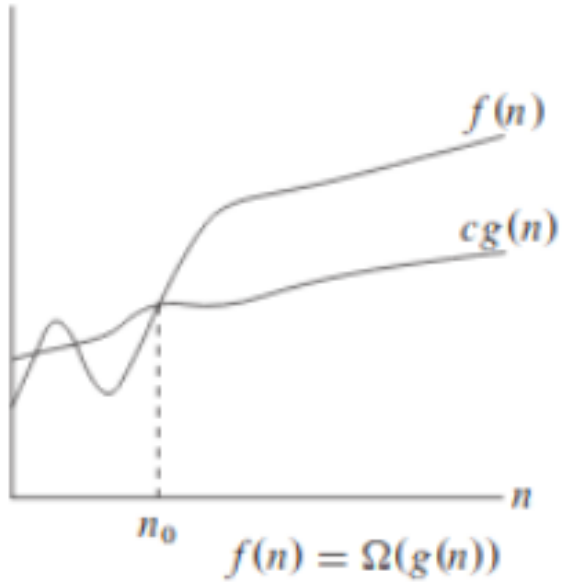
Notacion O-grande

Ejemplo 2.4. *Probar que* $an + b = O(n^2)$ *para todo* $a > 0$.

Notacion Omega

Dada una función $g(n)$, definimos $\Omega(g(n))$ según

$$\Omega(g(n)) = \{f(n) : \text{existen constantes positivas } c, n_0 \text{ tales que } 0 \leq cg(n) \leq f(n) \text{ para todo } n \geq n_0\}$$



Notacion Theta

Dada una función $g(n)$, definimos $\Theta(g(n))$ según

$$\Theta(g(n)) = \{f(n) : \text{existen constantes positivas } c_1, c_2, n_0$$
$$\text{tales que } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ para todo } n \geq n_0\}.$$

Notacion Theta

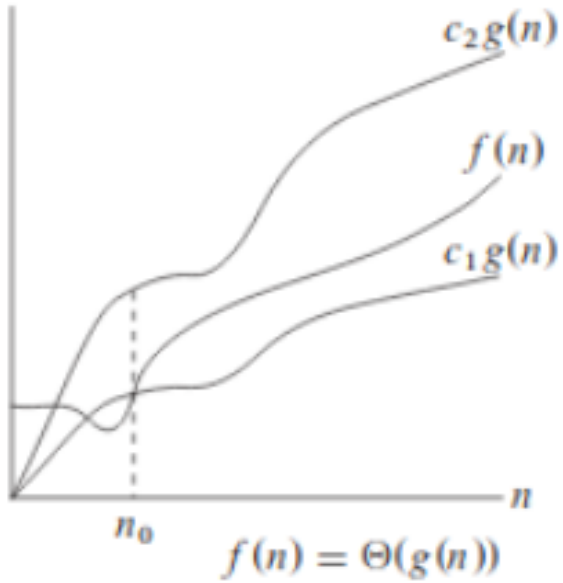


Figure 4: Tomada del libro Cormen, Introduction to Algorithms

Notacion Theta

Ejemplo 2.1. *Demostrar que* $\frac{1}{2}n^2 - 3n = \Theta(n^2)$.

Notacion Theta

Ejemplo 2.2. *Demostrar que* $6n^3 \neq \Theta(n^2)$.

Notacion Theta

Ejercicio 2.1. $an^2 + bn + c = \Theta(n^2)$ *para todo* $a > 0$.

Notacion o

Dada una función $g(n)$, definimos $o(g(n))$ según

$$o(g(n)) = \{f(n) : \text{para cada constante } c > 0$$

$$\text{existe una constante } n_0 \text{ tal que } 0 \leq f(n) < cg(n) \text{ para todo } n \geq n_0\}$$

Notacion o

Ejemplo 2.7. $2n = o(n^2)$

Notacion o

Ejemplo 2.8. $2n^2 \neq o(n^2)$

Notacion o

Observación 2.4. $f(n) = o(g(n))$ *si y solo si* $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

Notacion ω

Dada una función $g(n)$, definimos $\omega(g(n))$ según

$$\omega(g(n)) = \{f(n) : \text{para cada constante } c > 0$$
$$\text{existe una constante } n_0 \text{ tal que } 0 \leq cg(n) < f(n) \text{ para todo } n \geq n_0\}$$

Propiedades

Transitividad

- $f(n) = \Theta(g(n))$, $g(n) = \Theta(h(n))$, entonces $f(n) = \Theta(h(n))$
- $f(n) = O(g(n))$, $g(n) = O(h(n))$, entonces $f(n) = O(h(n))$
- $f(n) = \Omega(g(n))$, $g(n) = \Omega(h(n))$, entonces $f(n) = \Omega(h(n))$
- $f(n) = o(g(n))$, $g(n) = o(h(n))$, entonces $f(n) = o(h(n))$
- $f(n) = \omega(g(n))$, $g(n) = \omega(h(n))$, entonces $f(n) = \omega(h(n))$

Propiedades

Reflexividad

- $f(n) = \Theta(f(n))$
- $f(n) = O(f(n))$
- $f(n) = \Omega(f(n))$

Propiedades

Simetría

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

Simetría transpuesta

- $f(n) = O(g(n))$ si y solo si $g(n) = \Omega(f(n))$
- $f(n) = o(g(n))$ entonces $g(n) = \omega(f(n))$

Propiedades

Observación 2.6. *Existen funciones no comparables, por ejemplo* n *y* $n^{1+\sin n}$.

Gracias