Running Times

CS3026 – Analysis & Design of Algorithms

Angel Napa

UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Content

1. Insertion Sort
2. O-Notation, Ω-Notation, Θ- Notation
3. o-Notation, ω-Notation

1 Insertion Sort

Insertion Sort

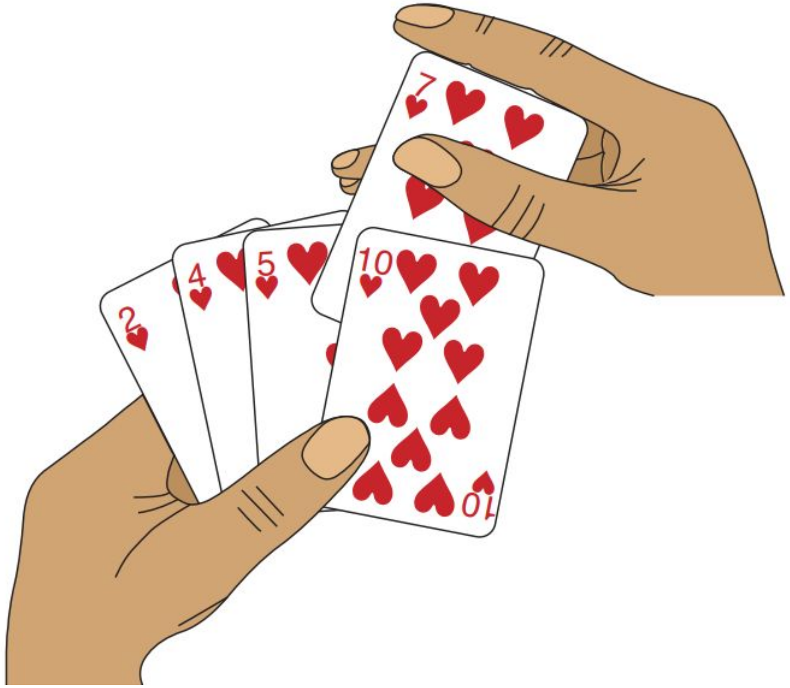

Figure 1: Cormen, Introduction to Algorithms

Insertion Sort

```
INSERTION-SORT(A, n)
for i = 2 to n
    key = A[i]
    // Insert A[i] into the sorted subarray A[1 : i − 1].
    j = i − 1
    while j > 0 and A[j] > key
        A[j + 1] = A[j]
        j = j − 1
    A[j + 1] = key
```

Figure 2: Cormen, Introduction to Algorithms

Insertion Sort

Invariant: At the start of each iteration of lines 1–8, the subarray $A[1 \ldots i-1]$ consists of the elements originally in $A[1 \ldots i-1]$ but in sorted order.

Insertion Sort

INSERTION-SORT(A, n)	*cost*	*times*
for $i = 2$ **to** n	c_1	n
$\quad key = A[i]$	c_2	$n-1$
$\quad$ // Insert $A[i]$ into the sorted subarray $A[1:i-1]$.	0	$n-1$
$\quad j = i - 1$	c_4	$n-1$
$\quad$ **while** $j > 0$ and $A[j] > key$	c_5	$\sum_{i=2}^{n} t_i$
$\quad\quad A[j+1] = A[j]$	c_6	$\sum_{i=2}^{n} (t_i - 1)$
$\quad\quad j = j - 1$	c_7	$\sum_{i=2}^{n} (t_i - 1)$
$\quad A[j+1] = key$	c_8	$n-1$

Figure 3: Cormen, Introduction to Algorithms

2

Asymptotic Notations
O - Notation

O - Notation

$$O(g(n)) = \{f(n) : \text{there exist constants } c, n_0$$
$$\text{such that } 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0\}$$

O - Notation

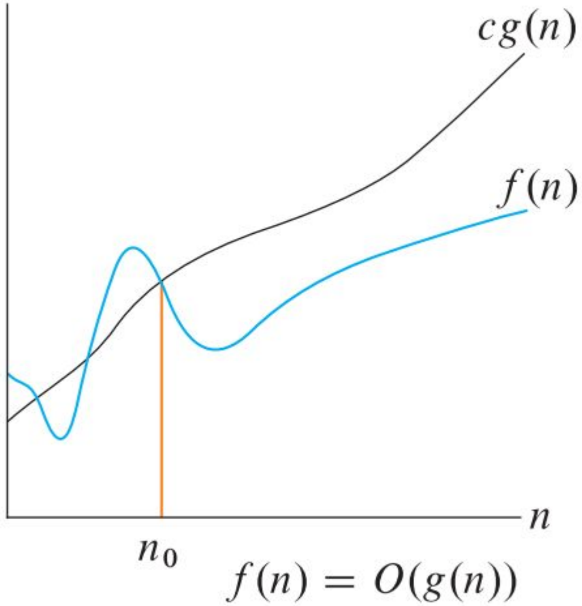

Figure 4: Cormen, Introduction to Algorithms

$$O(g(n)) = \{f(n) : \text{there exist constants } c, n_0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

Exercises

O - Notation

Example 2.1. *Show that*

$$n^2 + 10n + 2 = O(n^2)$$

O - Notation

Example 2.2. *Show that*

$$\frac{n^2}{2} + 3n = O(n^2)$$

O - Notation

Example 2.3. *Show that $n/100$ is not $O(1)$.*

O - Notation

Example 2.4. *Show that* $an + b = O(n^2)$ *for all* $a > 0$.

3 Asymptotic Notations
Ω - Notation

O - Notation

$$\Omega(g(n)) = \{f(n) : \text{there exist constants } c, n_0$$
$$\text{such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0\}$$

Ω - Notation

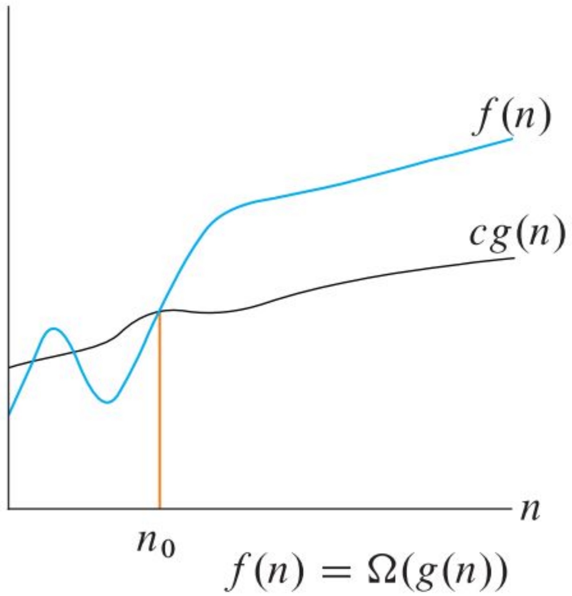

Figure 4: Cormen, Introduction to Algorithms

$$\Omega(g(n)) = \{f(n) : \text{there exist constants } c, n_0 \text{ such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0\}$$

Θ - Notation

$$\Theta(g(n)) = \{f(n) : \text{there exist constants } c_1, c_2, n_0$$
$$\text{such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0\}.$$

Θ - Notation

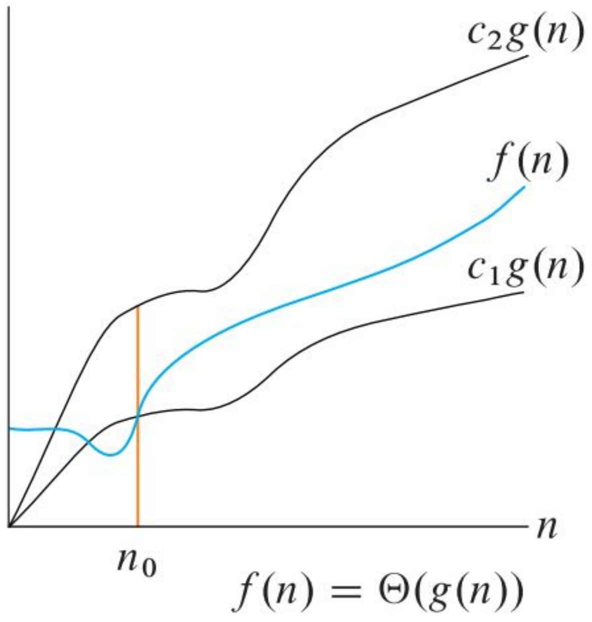

Figure 4: Cormen, Introduction to Algorithms

$$\Theta(g(n)) = \{f(n) : \text{there exist constants } c_1, c_2, n_0 \text{ such that } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for all } n \ge n_0\}.$$

Exercises

Θ - Notation

Example 2.5. *Show that* $\frac{1}{2}n^2 - 3n = \Theta(n^2)$.

Θ - Notation

Example 2.6. *Show that* $6n^3 \neq \Theta(n^2)$.

Θ - Notation

Exercise 2.1. $an^2 + bn + c = \Theta(n^2)$ *for all* $a > 0$.

4 Asymptotic Notations
o - Notation

o - Notation

$$o(g(n)) = \{f(n) : \text{for each constant } c > 0$$

$$\text{there is a constant } n_c \text{ such that } 0 \leq f(n) < cg(n) \text{ para todo } n \geq n_c\}$$

Exercises

o - Notation

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

o - Notation

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

o - Notation

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

ω - Notation

Given a function $g(n)$, we define $\omega(g(n))$ as

$$\omega(g(n)) = \{f(n) : \text{for each constant } c > 0$$
$$\text{there exists a constant } n_c \text{ such that } 0 \leq cg(n) < f(n) \text{ for all } n \geq n_c\}$$

Properties

Transitivity

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

Properties

Reflexivity

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

Properties

Symmetry

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

Transpose symmetry

- $f(n) = O(g(n))$ if and only if $g(n) = \Omega(f(n))$
- $f(n) = o(g(n))$ if and only if $g(n) = \omega(f(n))$

Observation

Observation 2.6. *There are functions that are not comparable, such as* n *and* $n^{1+\sin n}$.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

> Reinventa el mundo <

Thank you!

Let’s keep finding mistakes!

TRANSFORMATEC