Universidad del Valle de Guatemala

Facultad de Ingeniería

Ciencias de la Computación y Tecnologías de la Información

Laboratorio 8

Teoría de la computación

José Luis Gramajo Moraga, Carné 22907

15 de octubre de 2025, Guatemala, Guatemala

Repositorio: https://github.com/Abysswalkr/TC-Lab08-Complejidad.git

Problema 1

$\# i = n - \lfloor n/2 \rfloor + 1$

$n = 2m$: $\# i = 2m - m + 1 = m + 1 = \frac{n}{2} + 1$

$\Rightarrow \# i = \Theta(n)$

$n = 2m+1$: $\lfloor n/2 \rfloor = m \Rightarrow \# i = (2m+1) - m + 1 = m + 2 = \frac{n+3}{2}$

$j \leq n - \frac{n}{2} = n - \lfloor n/2 \rfloor = \lceil n/2 \rceil$

$\# j = \lceil n/2 \rceil = n - \lfloor n/2 \rfloor$

$n = 2m$: $\# j = m = \frac{n}{2}$

$\Rightarrow \# j = \Theta(n)$

$n = 2m+1$: $\# j = m + 1 = \frac{n+1}{2}$

$\# k = \lfloor \log_2 n \rfloor + 1 \quad (n \geq 1)$ $\quad$ $n=0$ el bucle no corre y $\# k = 0$

$\# k = \Theta(\log n)$

$T(n) = \underbrace{(n - \lfloor n/2 \rfloor + 1)}_{\# i} \underbrace{(n - \lfloor n/2 \rfloor)}_{\# j} \underbrace{(\lfloor \log_2 n \rfloor + 1)}_{\# k}$

$n = 10$:

$\# i = 10 - \lfloor 10/2 \rfloor + 1 = 6 \quad (i = 5, 6, 7, 8, 9, 10)$

$\# j = \lceil 10/2 \rceil = 5 \quad (j = 1 \ldots 5$ porque $j + 5 \leq 10)$

$\# k = \lfloor \log_2 10 \rfloor + 1 = 4 \quad (k = 1, 2, 4, 8)$

$T(10) = 6 \cdot 5 \cdot 4 = 120$

$n - \lfloor n/2 \rfloor \geq n/2$, $\quad n - \lfloor n/2 \rfloor + 1 \geq n/2$, $\lfloor \log_2 n \rfloor + 1 \geq \log_2 n$

$\Rightarrow T(n) \geq n/2 \, (n/2)(\log_2 n) = 1/4 \, n^2 \log_2 n \Rightarrow T(n) \in \Omega(n^2 \log n)$.

Problema 2

$n > 1 \Rightarrow T(n) = \Theta(1)$

$n > 2 \Rightarrow i = 1 \ldots n \Rightarrow \# i = n$

$j \Rightarrow \# j = 1$

$n \geq 2$: $\quad T(n) = \# i \cdot \# j = n \cdot 1 = n$

Big Oh: $\quad T(n) \leq 1 \cdot n \Rightarrow T(n) \in O(n)$

Cada i realiza al menos 1 operación útil $\Rightarrow T(n) \geq cn$ para $n \geq 2 \Rightarrow T(n) = \begin{cases} \Theta(1), & n \leq 1 \\ \Theta(1), & n \geq 2 \end{cases} \Rightarrow T(n) \in \Theta(n)$

Problema 3

$i : i = 1 \ldots \lfloor n/3 \rfloor \Rightarrow \#i = \lfloor n/3 \rfloor$

$j : j = 1, 5, 9, \cdots \leq n \Rightarrow \#j = \lceil n/4 \rceil$

$T(n) = \lfloor n/3 \rfloor \cdot \lceil n/4 \rceil$

$T(n) \leq n/3 \, (n/4 + 1) \leq c_2 n^2 \Rightarrow T(n) \in O(n^2)$

$T(n) \geq (n/3 - 1)(n/4) \geq c_1 n^2 = T(n) \in \Omega(n^2)$

$T(n) \in \Theta(n^2)$

Problema 4

¿Cuál es el mejor?

- **Para una búsqueda en arreglo ya ordenado:**
  Mejor = **Búsqueda Binaria** ($\Theta(\log\, n)$ peor/promedio) $\ll$ Lineal ($\Theta(n)$).
- **Para una sola búsqueda en arreglo no ordenado (sin querer ordenar):**
  Mejor = **Búsqueda Lineal**.
  (Ordenar para luego hacer Binaria cuesta $\Omega(n\log\, n)$, peor que mirar una sola vez $\Theta(n)$.)
- **Para muchas búsquedas sobre el mismo arreglo no ordenado (m consultas):**
  Ordenar una vez (Quick Sort, $\approx n\log\, n$) y luego buscar con Binaria ($m\log\, n$).
  Comparación:

$$\text{Lineal total} \approx m\, n \text{vsOrdenar+Binaria} \approx n\log\, n + m\log\, n.$$

Ordenar+Binaria es mejor cuando

$$m\, n\ \>;\ n\log\, n + m\,\log n \Rightarrow\ m\ \>;\ \frac{\text{n}\log n}{n - \log n}\ \approx\ \log n.$$

- **Para ordenar (no para buscar):**
  Quick Sort es **mejor en promedio** $\Theta(n\log\, n)$(baja constante y buen uso de caché), pero su **peor caso** es $\Theta(n^2)$; con pivote aleatorio/median-of-three se evita casi siempre ese peor caso.

1) Linear Serch: arreglo de tamaño n, sin orden

Mejor caso:

$$C(n) = 1 \Rightarrow \Theta(1)$$

Peor caso:

$$C(n) = n = \Theta(n)$$

Caso promedio:

$$E[C] = \frac{1+2+\cdots+n}{n} = \frac{n+1}{2}$$

$$E[C] = p \cdot \frac{n+1}{2} + (1-p)\cdot n = n - \frac{p}{2}(n-1) \Rightarrow \Theta(n)$$

2) Binary Search: arreglo ordenado de tamaño n

Mejor caso:

$$C(n) = 1 \Rightarrow \Theta(1)$$

Peor caso:

$$T(n) = T(\lfloor n/2 \rfloor) + 1,\ T(1) = 1 \Rightarrow C(n) = \lfloor \log_2 n \rfloor + 1 \Rightarrow \Theta(\log n)$$

3) Quick sort: participación lineal por nivel

Mejor caso:

$$T(n) = 2T(n/2) + cn \Rightarrow T(n) = \Theta(n \log n)$$

Caso promedio:

$$T(n) = 1/n \sum_{k=0}^{n-1} (T(k) + T(n-1-k)) + cn$$

Peor caso:

$$T(n) = T(n-1) + T(0) + cn \Rightarrow C(n) = \frac{n(n-1)}{2} \Rightarrow \Theta(n^2)$$

Problema 5

Inciso a: Verdadero

$\exists c_1, c_2 > 0,\ n_0 : \ c_1 g(n) \le f(n) \le c_2 g(n) \quad (n \ge n_0)$

$\exists d_1, d_2 > 0,\ n_1 : \ d_1 h(n) \le g(n) \le d_2 h(n) \quad (n \ge n_1)$

$n \ge N = \max\{n_0, n_1\}$

$c_1 d_1 h(n) \le f(n) \le c_2 d_2 h(n)$

$\Rightarrow f(n) = \Theta(h(n))$. Como Θ es simétrica $h(n) = \Theta(f(n))$

Inciso b: Verdadero

$\exists C > 0, n_0 : \ f(n) \le C g(n), \qquad \exists D > 0, n_1 : \ g(n) \le D h(n)$

$\Rightarrow f(n) \le CD h(n) \quad (n \ge N = \max\{n_0, n_1\})$

$f(n) = O(h(n)) \rightarrow \ f = O(h) \Leftrightarrow h = \Omega(f)$

$h(n) \ge \frac{1}{CD} f(n) \Rightarrow h(n) = \Omega(f(n))$

Inciso c: Falso $\quad f(n) = \Theta(n^3)$

1. tuple (range $(0, n)$) $\rightarrow O(n)$
2. Doble bucle: pares (i, j) con $0 \le i < j \le n-1$
   - En cada par se crea el slice "atupla[i:j] de longitud $k = j - i$ y se hashea para insertar en el set.
   - Coste por iteración: $O(k) = O(j - i)$
3. Suma total:

$$T(n) = \sum_{i=0}^{n-1} \sum_{j=i+1}^{n-1} (j - i) = \sum_{k=1}^{n-1} (n-k)k = n \sum_{k=1}^{n-1} k - \sum_{k=1}^{n-1} k^2 = n \frac{(n-1)n}{2} - \frac{(n-1)n(2n-1)}{6} = \frac{(n-1)n(n+1)}{6} = \Theta(n^3)$$

4. El termino $O(n)$ no cambia el orden: $f(n) = \Theta(n^3)$

Conclusión:

- 5.a **Verdadero**
- 5.b **Verdadero**
- 5.c **Falso**; el tiempo es $\Theta(n^3)$ no $\Theta(n^2)$.