Universidad Nacional Autónoma de México

Facultad de Ciencias

Tarea 2

Análisis de Algoritmos

Información del curso	
Profesor	María de Luz Gasca Soto
Ayudante	Brenda Margarita Becerra Ruíz
Ayudante	Enrique Ehecatl Hernández Ferreiro

Autor: Juárez Torres Carlos Alberto

```
______________1¶¶¶____¶¶¶1___¶¶¶1_________________
_____________¶¶¶1___1¶¶1___1¶¶1___________________
____________1¶¶1___1¶¶1___1¶¶1____________________
____________1¶¶1___1¶¶1___1¶¶¶____________________
_____________¶¶¶____¶¶¶1___¶¶¶1___________________
______________¶¶¶¶___1¶¶¶___1¶¶¶__________________
_______________1¶¶¶1___¶¶¶1___¶¶¶¶________________
_________________1¶¶1____¶¶¶____¶¶¶_______________
___________________¶¶1____¶¶1____¶¶1______________
___________________¶¶¶____¶¶¶____¶¶¶______________
__________________1¶¶1___1¶¶1____¶¶1______________
_________________¶¶¶____¶¶¶1___1¶¶1_______________
________________11_____111_____11_________________ Toma un café mañanero UwU
__________¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
1¶¶¶¶¶¶¶¶¶¶¶__¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
1¶¶¶¶¶¶¶¶¶¶¶__1¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
1¶¶_______¶¶__1¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
1¶¶_______¶¶__1¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
1¶¶_______¶¶__¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
1¶¶_______¶¶__1¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
_¶¶¶¶¶¶¶¶¶¶¶__¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
_¶¶¶¶¶¶¶¶¶¶¶__¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶________
__________¶¶___1¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶1________
__________1¶¶___¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶_________
____________¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶11__________
11_____________________________________________111
1¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶1
__¶¶111111111¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶¶111111111¶__
```

Fecha de entrega: February 24, 2023

1 Sea P un problema. El desempeño computacional en el peor de los casos para P es $O(n^2)$ y tambien es $\Omega(n \log_2 n)$ sea A un algoritmo que soluciona a P. ¿Cuáles de las siguientes afirmaciones resultan consistentes con la información sobre P?

Las afirmaciones consistentes con la información sobre P son:

- $\Theta(n^2)$ es consistente con la información sobre P, ya que se sabe que el desempeño computacional en el peor de los casos para P es $O(n^2)$
- $\Theta(n \log n)$ es consistente con la información sobre P, ya que se sabe que el desempeño computacional en el peor de los casos para P es $\Omega(n \log n)$
- Sobre $O(n^{3/2})$, $O(n)$, $O(n^3)$ no son consistentes con la información de P, ya que ninguna de ellas está incluida en el rango de complejidad de P en el peor de los casos para P

2 Proporcionar un problema que satisfaga la Hipótesis del Ejercicio 1 y que cumpla al menos tres de los cinco incisos. Debe indicar el problema y enunciar los algoritmos.

El problema es tan simple como el de la ordenación de números y los algoritmos que tomaremos en cuestión

- Bubble Short. En el peor de los casos su desempeño computacional es de $O(n^2)$
- Merge Short. En el peor de los casos su desempeño computacional es de $\Theta(n \log n)$
- Quick Short. En el peor de los casos su desempeño computacional es de $O(n^2)$, aunque en la practica generalmente se tiene el desempeño computacional de $\Theta(n \log n)$

3 Usando la definición formal de O, o, ω y Ω justificar si los siguientes argumentos son correctos o dar un contraejemplo.

- $2n \in O(n^2)$ & $2n^2 \notin o(n^2)$:
  El argumento es correcto. Para justificarlo, podemos utilizar la definición formal de la notación o (pequeña o).

  Decimos que $f(n) \in o(g(n))$ si existe una constante $c > 0$ tal que para todo n suficientemente grande, se cumple que $f(n) \leq c * g(n)$. En este caso, podemos tomar $c = 1$ y vemos que $2n \leq n^2$ para $n \geq 2$, lo cual implica que $2n \in o(n^2)$.

Por otro lado, decimos que $f(n) \notin o(g(n))$ si para cualquier constante $c > 0$, no existe un valor de n suficientemente grande tal que $f(n) \leq c * g(n)$. En este caso, podemos tomar $c = 1$ y ver que $2n^2 > n^2$ para todo $n \geq 1$, lo que implica que $2n^2$ no pertenece a $o(n^2)$. Por lo tanto, el argumento es correcto.

- $2n$ es $\Omega(5 \log n)$
  Decimos que $f(n)$ es $\Omega(g(n))$ si existe una constante $c > 0$ y un valor n_0 tal que para todo $n \geq n_0$, se cumple que $f(n) \geq c \cdot g(n)$. En este caso, podemos tomar $c = 1$ y ver que $2n \geq 5 \log n$ para todo $n \geq 2$. Sin embargo, la definición de Ω también requiere que exista un valor n_0 tal que $f(n) \geq c \cdot g(n)$ para todo $n \geq n_0$, y esto no es cierto en este caso.

  Para demostrar esto, podemos utilizar el límite de la razón. La razón entre $2n$ y $5 \log n$ tiende a infinito cuando n tiende a infinito. Esto significa que no hay un valor de n_0 tal que $2n \geq c \cdot 5 \log n$ para todo $n \geq n_0$ y cualquier constante $c > 0$. Por lo tanto, el argumento es incorrecto.

- $\log 3^n$ es $\Omega(\log 4^n)$
  El argumento es correcto. Para justificarlo, podemos utilizar la definición formal de la notación Ω (Omega).

  Decimos que $f(n)$ es $\Omega(g(n))$ si existe una constante $c > 0$ y un valor n_0 tal que para todo $n \geq n_0$, se cumple que $f(n) \geq c \cdot g(n)$. En este caso, podemos utilizar la propiedad de logaritmos que dice que $\log a^n = n \cdot \log a$ para cualquier base a. Entonces, $\log 3^n = n \cdot \log 3$ y $\log 4^n = n \cdot \log 4$.

  Podemos tomar $c = 1/2$ y ver que $\log 3^n \geq (1/2) \cdot \log 4^n$ para todo $n \geq 1$. Esto se puede demostrar tomando logaritmo natural en ambos lados de la desigualdad:

  $$\log 3^n \geq (1/2) \cdot \log 4^n$$

  $$n \cdot \log 3 \geq (1/2) \cdot n \cdot \log 4$$

  $$\log 3 \geq (1/2) \cdot \log 4$$

  $$\log 3^2 \geq \log 4$$

  $$2 \cdot \log 3 \geq \log 4$$

  Esta última desigualdad es cierta porque $\log 3$ es aproximadamente 1.585 y $\log 4$ es aproximadamente 1.386, por lo que $2 \cdot \log 3 > \log 4$. Por lo tanto, el argumento es correcto.

- 5^n es $O(3^n)$
  El argumento es incorrecto. Para justificarlo, podemos utilizar la definición formal de la notación O (grande o).

  Decimos que $f(n)$ es $O(g(n))$ si existe una constante $c > 0$ y un valor n_0 tal que para todo $n \geq n_0$, se cumple que $f(n) \leq c \cdot g(n)$. En este caso, podemos tomar $c = 5$ y ver que $5^n \leq 5 \cdot 3^n$ para todo $n \geq 1$.

Sin embargo, la definición de O también requiere que exista un valor n_0 tal que $f(n) \leq c \cdot g(n)$ para todo $n \geq n_0$, y esto no es cierto en este caso. Podemos demostrar esto tomando el límite de la razón:

$$\lim_{n\to\infty} \frac{5^n}{3^n} = \lim_{n\to\infty} \left(\frac{5}{3}\right)^n = \infty$$

Esto significa que no hay un valor de n_0 tal que $5^n \leq c \cdot 3^n$ para todo $n \geq n_0$ y cualquier constante $c > 0$. Por lo tanto, el argumento es incorrecto.

- Si $f(n) = 4n^2 \log n + 7n^2 + 10n$
  Podemos analizar la complejidad asintótica de la función $f(n)$ mediante la notación O (grande o). Para esto, necesitamos encontrar una función $g(n)$ tal que $f(n) \leq c \cdot g(n)$ para una constante $c > 0$ y n suficientemente grande.

  Podemos observar que el término dominante en la función $f(n)$ es $4n^2 \log n$. Por lo tanto, podemos elegir $g(n) = n^2 \log n$ y $c = 5$, por ejemplo. Entonces, para $n \geq 1$, se cumple:

  $$\begin{aligned} f(n) = 4n^2 \log n + 7n^2 + 10n &\leq 4n^2 \log n + 7n^2 \log n + 10n^2 \\ = (4 + 7 + 10)n^2 \log n &= 21n^2 \log n \leq 5n^2 \log n = c \cdot g(n) \end{aligned}$$

  Por lo tanto, concluimos que $f(n) \in O(n^2 \log n)$. Esto significa que la función $f(n)$ tiene una complejidad asintótica no peor que $n^2 \log n$, es decir, que $f(n)$ crece a una tasa no mayor que $n^2 \log n$ para valores suficientemente grandes de n.

- Si $f(n) = 4n^2 \log n + 7n^2 + 10n$ entonces $f(n) \in \Theta(n^3)$ Para verificar si $f(n) \in \Theta(n^3)$, debemos demostrar que existen constantes c_1, c_2 y n_0 tales que se cumpla:

  $$c_1 n^3 \leq f(n) \leq c_2 n^3 \quad \forall n \geq n_0.$$

  Empecemos por la primera desigualdad:

  $$4n^2 \log n + 7n^2 + 10n \geq c_1 n^3$$

  Dividimos ambos lados por n^3:

  $$\frac{4}{\log n} + \frac{7}{n} + \frac{10}{n^2} \geq c_1$$

  El lado izquierdo de la desigualdad anterior tiende a 0 cuando n se acerca a infinito. Por lo tanto, no podemos encontrar una constante c_1 que satisfaga la primera desigualdad para todo $n \geq n_0$.

  Por lo tanto, la afirmación "$f(n) \in \Theta(n^3)$" es falsa. En otras palabras, $f(n)$ no está acotada superior e inferiormente por una función de la forma cn^3 para todo n suficientemente grande.

4 Resuelve con sumo detalle uno de los siguiente ejercicios

4.1 Ejercicio elegido: El Algoritmo A realiza $30n^2$ y el algoritmo B ejecuta $500 \ln n$ operaciones elementales. ¿Para que valor de n el Algoritmo B empieza a mostrar un mejor desempeño?

Primero, establecemos una ecuación que iguale el tiempo de ejecución de ambos algoritmos:

$$30n^2 = 500 \ln n \tag{1}$$

Podemos resolver esta ecuación utilizando el método de Newton-Raphson o cualquier otro método numérico, pero en este caso podemos resolverla analíticamente para obtener una solución exacta.

Primero, dividimos ambos lados de la ecuación por n^2 y luego aplicamos la función exponencial a ambos lados:

$$e^{30/n^2} = e^{500 \ln n/n^2} \tag{2}$$

A continuación, tomamos la raíz cuadrada de ambos lados y simplificamos:

$$e^{15/n} = n^5 \tag{3}$$

Tomamos logaritmo natural en ambos lados:

$$\frac{15}{n} = 5 \ln n \tag{4}$$

Despejamos n:

$$n = \frac{e^{15/5}}{\ln(e^{15/5})} \approx 164.72 \tag{5}$$

Por lo tanto, para valores de n mayores a 164.72, el Algoritmo B tendrá un mejor desempeño que el Algoritmo A.

5 Clasificar las siguientes funciones de complejidad:

1. $6n^5 + 27$ Pertenece a $O(n^5)$, $\Omega(1)$, $\Theta(n^5)$ y $\omega(1)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.
2. $\frac{2n}{7} + \log n$

   Pertenece a $O(n)$, $\Omega(\log n)$, $\Theta(n)$ y $\omega(\log n)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.
3. $2n + \ln n$

   Pertenece a $O(n)$, $\Omega(\ln n)$, $\Theta(n)$ y $\omega(\ln n)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.
4. $3n \log \log n$

   Pertenece a $O(n \log \log n)$, $\Omega(n)$, $\Theta(n \log \log n)$ y $\omega(1)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.
5. $5 \ln n + 8$

   Pertenece a $O(\ln n)$, $\Omega(1)$, $\Theta(\ln n)$ y $\omega(1)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.
6. $5n^2 + n \log n$

   Pertenece a $O(n^2)$, $\Omega(n \log n)$, $\Theta(n^2)$ y $\omega(\log n)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.

7. $7n^3 + 3n^2 + 5n$

   Pertenece a $O(n^3)$, $\Omega(n^3)$, $\Theta(n^3)$ y $\omega(1)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.

8. $2^n + 8n^2 + 5n + \log\log n$

   Pertenece a $O(2^n)$, $\Omega(n^2)$, $\Theta(2^n)$ y $\omega(\log n)$, pero no pertenece a $o(n^2)$ ni a $\omega(n^2)$.

Table 1: Tabla de pertenencia de funciones

Conjunto de funciones	Funciones
O(n^2)	$6n^5$ + 27, $5n^2$ + n log n, $7n^3$ + $3n^2$ + 5n
$\Omega(n^2)$	2^n + $8n^2$ + 5n + log log n
$\Theta(n^2)$	Ninguna de las funciones
o(n^2)	Ninguna de las funciones
$\omega(n^2)$	3n log log n

6 Ejercicio Opcional

Supongamos que tenemos una computadora que requiere un minuto para resolver ejemplares de tamaño $n = 1000$. ¿Qué ejemplares pueden ser ejecutados en un minuto si compramos una nueva computadora mil veces más rápida que la anterior, suponiendo las siguientes complejidades $T(n)$?

1. Si $T(n)$ es $\Theta(n)$, entonces la computadora tarda n segundos en resolver un problema de tamaño n. Si la nueva computadora es mil veces más rápida, entonces tardará $n/1000$ segundos en resolver el mismo problema. Para resolver el problema en un minuto (60 segundos), podemos resolver problemas de tamaño $n \leq 60,000$. En otras palabras, la nueva computadora puede resolver problemas 60 veces más grandes que la computadora original.

2. Si $T(n)$ es $\Theta(n^3)$, entonces la computadora tarda n^3 segundos en resolver un problema de tamaño n. Si la nueva computadora es mil veces más rápida, entonces tardará $(n/10)^3$ segundos en resolver el mismo problema. Para resolver el problema en un minuto (60 segundos), podemos resolver problemas de tamaño $n \leq 215$. En otras palabras, la nueva computadora puede resolver problemas aproximadamente 5.5 veces más grandes que la computadora original.

3. Si $T(n)$ es $\Theta(10^n)$, entonces la computadora tarda 10^n segundos en resolver un problema de tamaño n. Si la nueva computadora es mil veces más rápida, entonces tardará 10^{n-3} segundos en resolver el mismo problema. Para resolver el problema en un minuto (60 segundos), podemos resolver problemas de tamaño $n \leq 3$. En otras palabras, la nueva computadora puede resolver problemas de tamaño pequeño, con n no mayor a 3.