UTEC

CIENCIA DE LA COMPUTACIÓN

Complejidad Algorítmica

CS1100 - Introducción a Ciencia de la Computación
Universidad de Ingeniería y Tecnología - UTEC

¿Qué tan eficiente es un algoritmo? - Tiempo

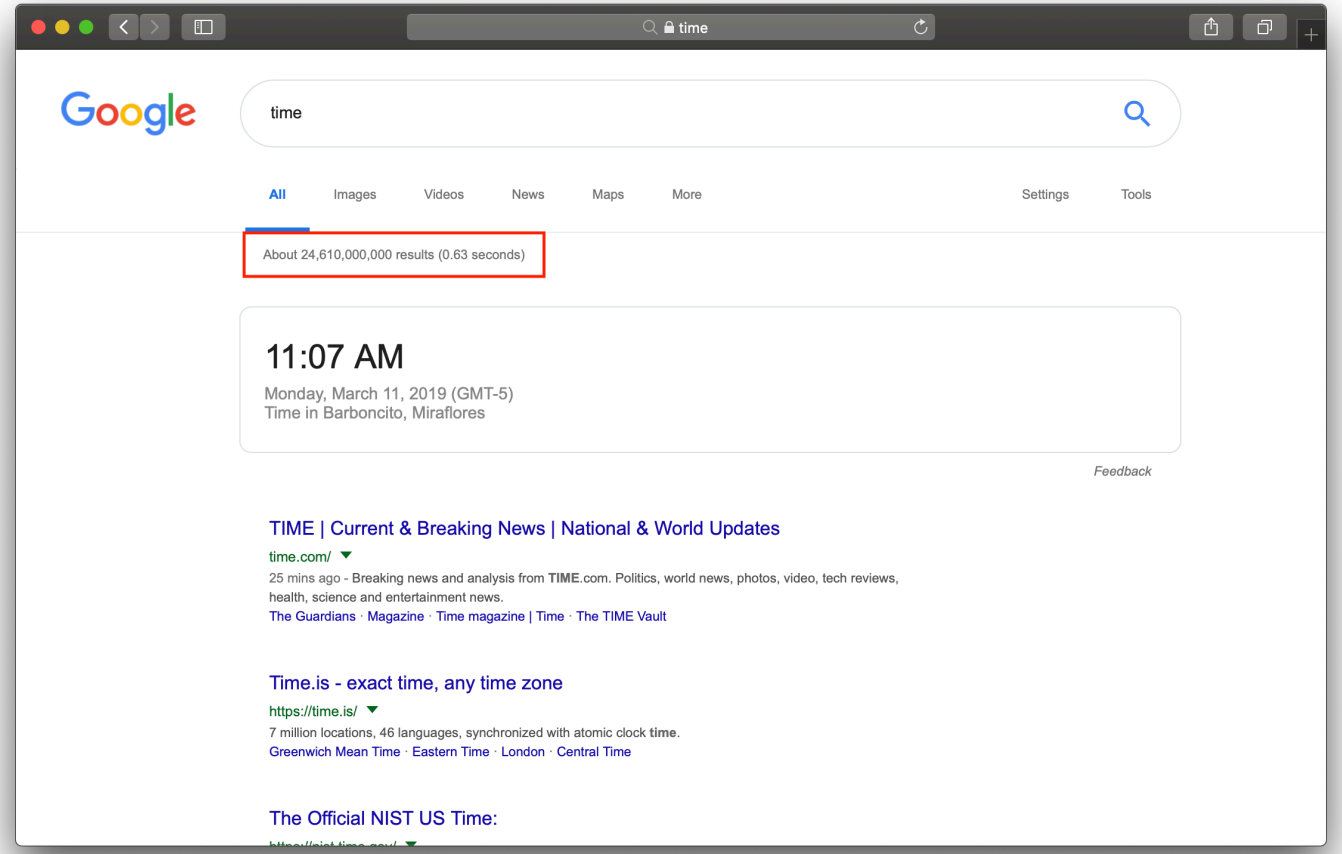


Figure: Google demora 0.63 segundos en obtener 24,610,000,000 resultados

¿Qué tan eficiente es un algoritmo? - Espacio

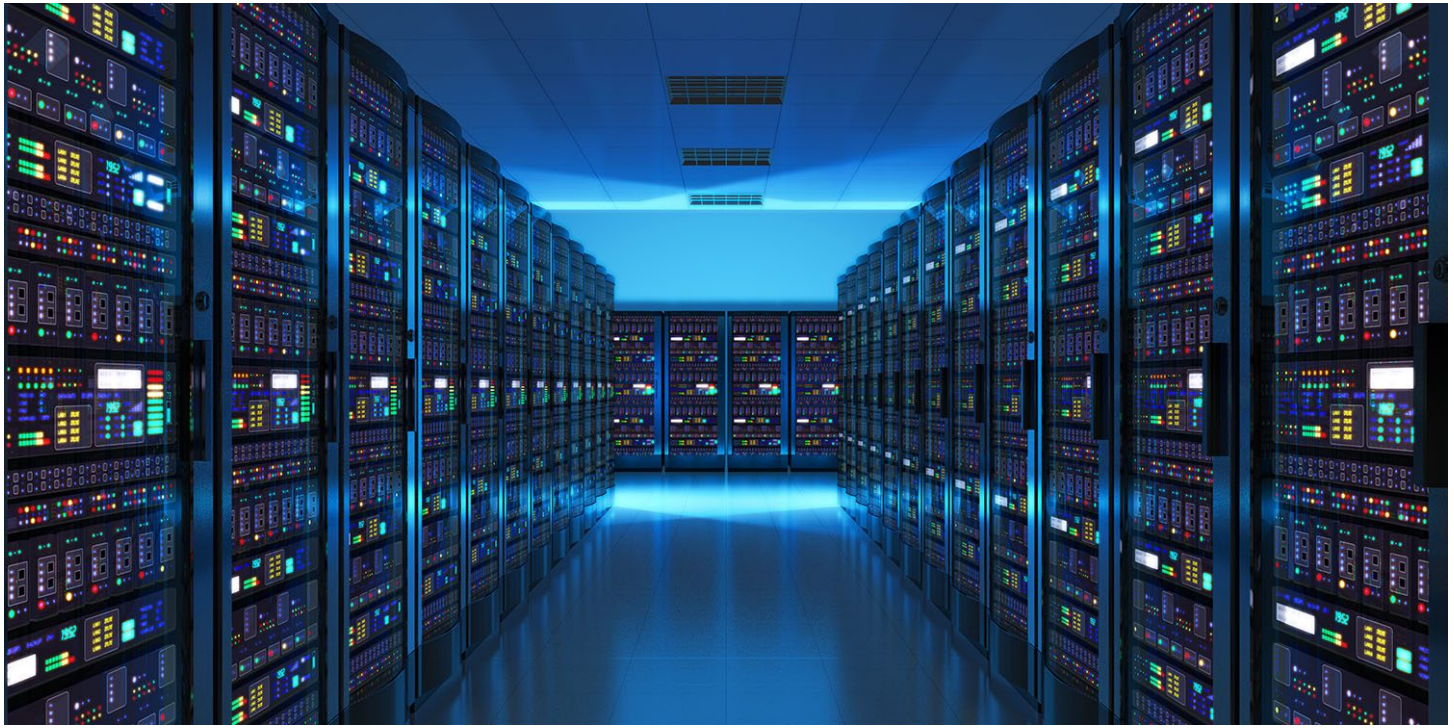


Figure: Google almacena ∼10-15 exabytes de información. Si estimamos que 1 computadora tiene 500GB, 1 exabyte equivaldría a 2 millones de computadoras

Si tu algoritmo demora 1 segundo para procesar una entrada de 1000 elementos, ¿cómo se comportará si duplicamos el número de elementos de entrada?

- ☐ Se demorará la misma cantidad de tiempo.
- ☐ Será el doble de rápido.
- ☐ Tomará 4 veces más tiempo.

¿ Por qué esto es importante ?

¿ De qué depende el análisis de un algoritmo ? ¿ Por qué ?

¿Cómo se mide el tiempo en Python?

```
import time # Librería para acceder a las variables de tiempo

start = time.time()

"""
Aquí van las líneas de código que van a ser medidas para
saber cuánto demoran en ejecutarse

"""

end          = time.time()
elapsed_time = end - start

print("It took %f seconds" % (elapsed_time))
```

¿Cómo se mide el tiempo en Python?

```
import time # Librería para acceder a las variables de tiempo

start = time.time()

n = 10
total = 0
for counter in range(1,n+1):
    total = total + counter

end          = time.time()
elapsed_time = end - start

print("Sum of 1 until %d is %d and took %f seconds" % (n, total, elapsed_time))
```

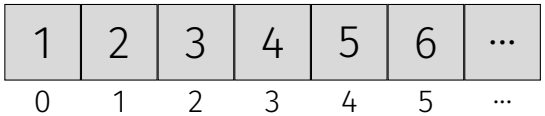
Arreglos en 1 dimensión


```python
# Crear un arreglo como lista
A = [10, 7, 3, 8, 5, 9]

# Crear un arreglo con NumPy
A = array([10, 7, 3, 8, 5, 9])

# Iterar usando el índice
for i in range(len(A)):
    print("Valor {}: {}".format(i + 1, A[i]))
```

¿ Qué información se puede representar usando arreglos en 1 dimensión ?

Arreglos en 1 dimensión - Ejercicio

Dado un arreglo de n números enteros consecutivos. ¿ Cuánto tiempo demora el algoritmo para obtener la sumatoria de los n números? Para:



- $n = 10^2$
- $n = 10^4$
- $n = 10^6$
- $n = 10^8$

Para verificar recuerda que la sumatoria de los n primeros números naturales está dada por $\frac{n(n+1)}{2}$

Arreglos en 2 dimensiones

	0	1	2	3	4
0	255	7	0	0	69
1	100	0	56	100	109
2	101	254	23	11	0
3	2	7	255	107	96
4	178	250	0	102	45

Figure: Matriz

```
# Crear una matriz usando listas
A = [[1, 4, 5],
     [-5, 8, 9]]

# Crear una matriz usando arreglos NumPy
A = np.array([[1, 2, 3], [3, 4, 5]])

# Iterar una matrix
for j in range(columns):
    for i in range(rows):
        print A[i][j]
```

Matrices o arreglos en 2 dimensiones

Arreglos en 2 dimensiones

	0	1	2	3	4
0	255	7	0	0	69
1	100	0	56	100	109
2	101	254	23	11	0
3	2	7	255	107	96
4	178	250	0	102	45

Figure: Matriz

- Resolver sistema de ecuaciones
- Representar imágenes
- Renderizar en pantallas
- Inteligencia Artificial

Aplicaciones en la vida real

Arreglos en 2 dimensiones

Figure: Imagen en escala de grises

- Resolver sistema de ecuaciones
- Representar imágenes
- Renderizar en pantallas
- Inteligencia Artificial

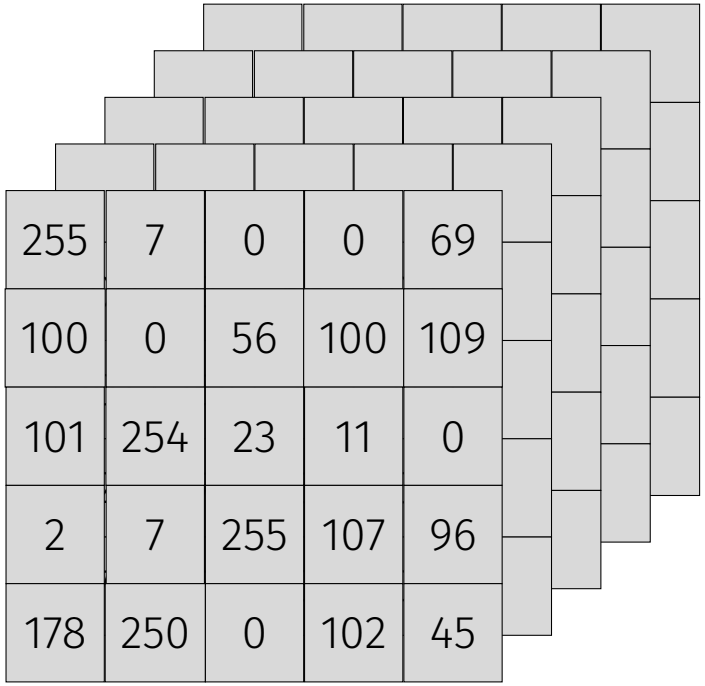
Aplicaciones en la vida real

Arreglos en 2 dimensiones - Ejercicio

Dada una matriz cuadrada M de tamaño $n \times n$, escriba el código de algunas funciones para medir el tiempo de ejecución de los siguientes problemas:

- Encontrar el elemento de menor valor
- Encontrar el elemento de mayor valor
- Encontrar la sumatoria de todos los valores
- Construir otra matriz con los elementos elevados al cuadrado.

Arreglos en 3 dimensiones



- Realidad virtual
- FaceID del iPhone
- Simulaciones quirúrgicas
- Gráficos de Fornite
- Física cuántica

Arreglos en 3 dimensiones

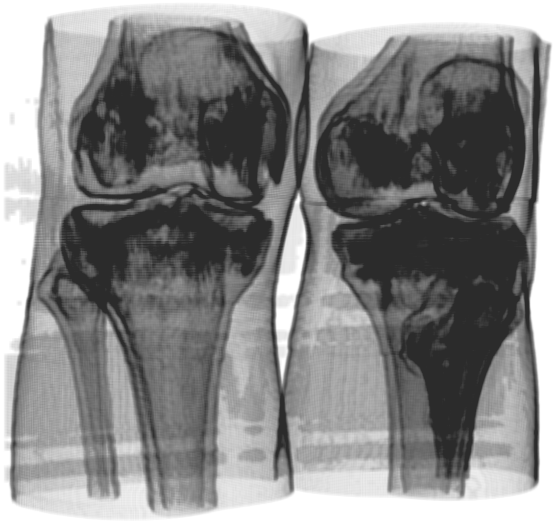


Figure: Imagen tomográfica

- Realidad virtual
- FaceID del iPhone
- Simulaciones quirúrgicas
- Gráficos de Fornite
- Física cuántica

Análisis Asintótico - Bucles anidados

```
sum = 0
for i = 1 to n do
    for j = 1 to n do
        sum = sum + 1
```

$$\sum_{i=1}^{n}\sum_{j=1}^{n} 1 = \sum_{i=1}^{n} n \\ = n^2 \tag{1}$$

```
sum = 0
for i = 1 to n do
    for j = i to n do
        sum = sum + 1
```

$$\sum_{i=1}^{n}\sum_{j=i}^{n} 1 = \sum_{i=1}^{n}(n-i+1) \\ = \sum_{i=1}^{n}(n+1) - \sum_{i=1}^{n} i \\ = n(n+1) - \frac{n(n+1)}{2} \\ = \frac{n(n+1)}{2} \\ = n^2 \tag{2}$$

Análisis Asintótico

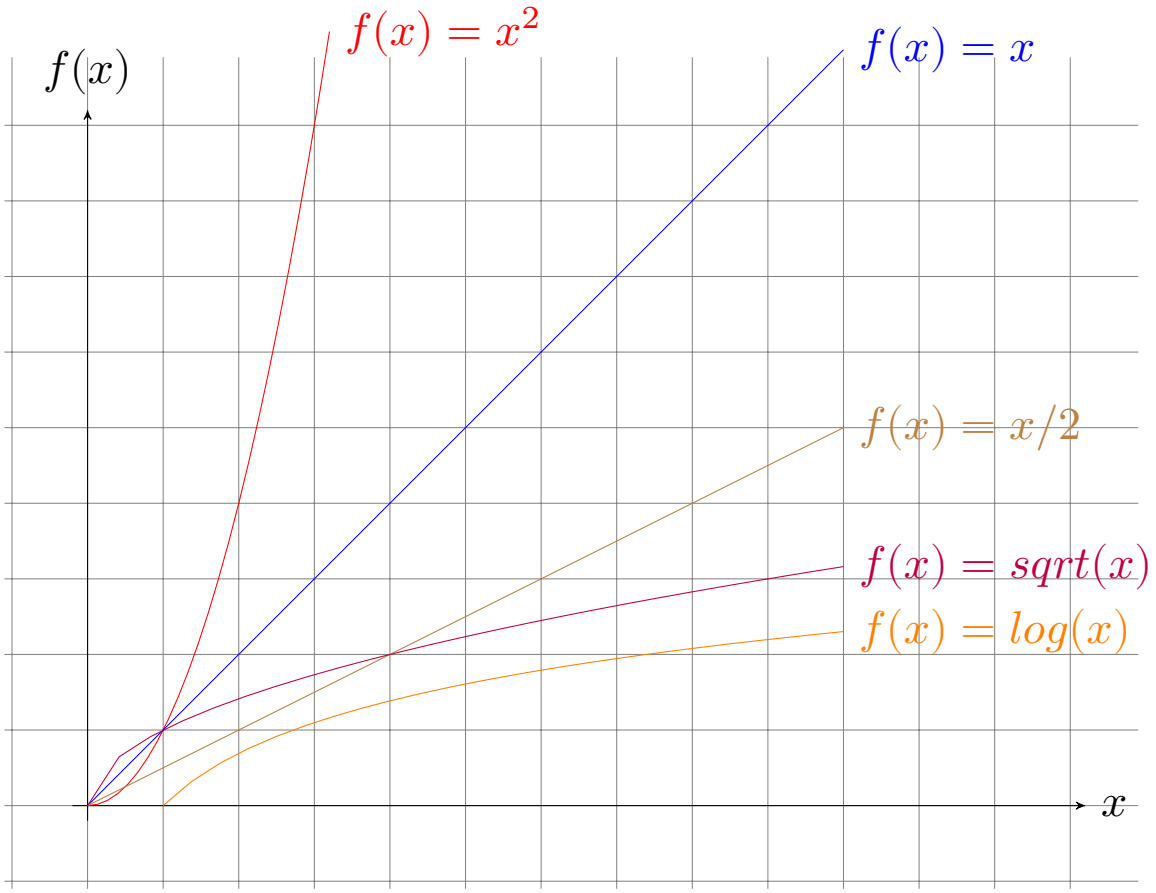


Figure: Análisis asintótico de algoritmos

Ejercicios y lectura adicional

Ejercicios

www.hackerrank.com
sem09-sesion-b-complejidad-01

Conteo de Vocales

www.hackerrank.com
sem09-sesion-b-complejidad-02

Rotación de matrices

www.hackerrank.com
sem09-sesion-b-complejidad-03

Rotación de matrices con repetición

www.hackerrank.com
sem09-sesion-b-complejidad-04

Suma de números

Lectura Adicional

Algorithm Analysis
@ahmedamedy/algorithm-analysis-bf0ca6506191

Resumen

El análisis de un algoritmo depende de la cantidad de elementos de entrada.

Empíricamente se puede medir el tiempo que demora un algoritmo para ser comparado.

Existe una manera teórica que permite comparar algoritmos denominado análisis asintótico.