Práctica 1.3: Complejidad teórica: algoritmos recursivos

Algoritmia y optimización

Curso 2024–25

1. Introducción

En esta Práctica 1.3 analizaremos la complejidad de algoritmos recursivos. Para los siguientes ejercicios, seguiremos practicando el análisis empírico.

2. Objetivos

- Comprender la complejidad teórica en algoritmos recursivos.
- Comparar la complejidad teórica con la complejidad empírica.
- Razonar los resultados obtenidos.

3. Ejercicio 1

Calcula la complejidad teórica del siguiente algoritmo.

```
función g(n):
    si n = 1:
        devuelve

    g(n-1)
    g(n-1)
```

Representa en un mismo gráfico las complejidades empírica y teórica. [1]

[1] Recuerda, como en la práctica anterior, que en la asignatura estamos utilizando la notación "Big O", que indica el orden de magnitud de la complejidad pero no su función exacta. Es por ello que, al dibujar en un mismo gráfico la complejidad empírica y la teórica, tendrás que aplicar algún factor para mantener la proporción de ambas curvas.

4. Ejercicio 2

Analiza la complejidad teórica y empírica del siguiente algoritmo que ya estudiamos en la Práctica 1_1. Visualiza con detalle el mejor y peor caso, así como el promedio, si los hubiera.

```
función ordena(arr):
    si |arr| <= 1:
        devuelve arr
    entonces:
        pivote = arr[0]
        para x hasta arr desde arr[1]:
            si x <= pivote:
                añadir elemento x en izq[]
            si x > pivote:
                añadir elemento x en der[]

        devuelve ordena(izq) + [pivote] + ordena(der)
```