Práctica 3
Diseño de Algoritmos

David Ruiz Rodríguez, Mohamed Rodrigo El Badry,
Nicolás Recinella Vidán, Denilson Palomino Adán

20 de diciembre de 2024

Índice

1. Introducción

Esta memoria contiene solo las respuestas a las preguntas teóricas. Si se quiere ver las respuestas que requieren código, entonces debe buscarlas en el código adjunto.

2. Ejercicio 1

Prescindiendo de la ciudad del Vaticano, ¿cuáles son las dos capitales de estados europeos más próximas entre sí?

Praga, de la República Checa y Bratislava, de Eslovaquia.

3. Ejercicio 4

Planteamos la separación de la nube de puntos según queden a la izquierda o a la derecha de cierta recta vertical, es decir, escogemos $x_0 \in \mathbf{Q}$ y adscribimos a P_1 los puntos (x, y) que cumplan $x < x_0$ y a P_2 los que cumplan $x > x_0$.

¿Siempre podemos encontrar $x_0 \in \mathbf{Q}$ de modo tal que este sistema reparta todos los puntos de la lista inicial en sublistas con $||P_1| - |P_2|| < 1$?

Asumiendo que todos los puntos tienen una coordenada x distinta, siempre se puede encontrar un x0 que divida los puntos en dos conjuntos iguales o en el peor caso con diferencia de uno. Esto se puede hacer tomando solo en cuenta la coordenada x, y ordenar los puntos respecto a esta y escoger el que está en el medio.

4. Ejercicio 6

Utilizando el teorema maestro, estima la complejidad del algoritmo en los casos siguientes:

$$T(n) = a \cdot T(\lfloor \frac{n}{b} \rfloor) + f(n)$$

$$\begin{cases} \text{Si } f(n) \in O(n^d), \text{ con } d < log_b a, & T(n) \in \Theta(n^{log_b a}); \\ \text{Si } f(n) \in \Theta(n^d), \text{ con } d = log_b a, & T(n) \in \Theta(n^{log_b a} \cdot log n); \\ \text{Si } f(n) \in \Omega(n^d), \text{ con } d > log_b a, & T(n) \in \Theta(f(n)). \end{cases}$$

En nuestro caso:

$$a = 1$$

$$b = 1$$

Por lo que:

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

  $d = 1$

  $O(n^1 \cdot log\ n) = O(n \cdot log\ n)$

- $f(n) = \Theta(n^2)$

  $d = 2$

  $O(n^2 \cdot log\ n)$

5. Ejercicio 11

Hemos abordado el problema de hallar la pareja (en rigor, una pareja) de puntos más próxima. ¿Cuál es la complejidad computacional de la búsqueda de la pareja más distante? Este otro problema, resuelve, en particular, el cálculo del diámetro del conjunto de entrada.

En base al algoritmo construido hasta ahora, sería $O(n \cdot log\ n)$, porque tendríamos que coger el mismo algoritmo que calcula la distancia mínima entre una pareja de puntos, y modificar algunas comparaciones. Como no modifica la complejidad del algoritmo, sería la misma.