ReflexAct4.3
Didier Aarón Ricardo Hernández Ferreira
A01663817

Cuando nos enfrentamos a la gestión de bases de datos extensas, optimizar el tiempo de procesamiento se vuelve fundamental. Un ejemplo claro se evidenció en nuestra primera tarea, donde implementamos diversos algoritmos y búsquedas utilizando vectores. Aunque estos algoritmos agilizan la búsqueda en listas de datos, el uso de vectores como estructuras de almacenamiento ralentiza significativamente el proceso. Por ejemplo, al procesar una base con casi 20 mil entradas, los programas podían tardar hasta 5 minutos en completar las operaciones. Esta demora puede deberse en parte a limitaciones de hardware o a la eficiencia del compilador, pero también se debe a la elección de vectores como medio de almacenamiento, lo cual dificulta el procesamiento.

Para superar estas limitaciones, incorporamos estructuras de datos más eficientes como grafos y árboles heap en nuestras tareas posteriores. Aunque en ocasiones aún empleamos vectores, el rendimiento general mejora notablemente gracias al uso de estas estructuras. Aunque requieren un manejo más complejo, son más eficientes para la computadora. Además, cada estructura cuenta con algoritmos específicos para ordenar y buscar datos, simplificando su manipulación y agilizando el procesamiento.

En nuestras clases, exploramos diversos tipos de estructuras de datos:

- Grafos: Estos nos permiten crear redes interconectadas, facilitando la búsqueda rápida de datos. Desarrollamos una clase llamada Graph para este propósito, que incluye métodos y atributos para acceder al grafo. Uno de los más relevantes es "shortestPath", que encuentra el camino más corto entre un nodo de origen y todos los demás nodos, lo que resulta útil para diversas aplicaciones.

- MaxHeap/Heap: Se trata de árboles binarios ordenados que nos permiten ordenar datos de manera eficiente. Creamos la clase MaxHeap para este fin, junto con las clases "ipAdress" e "IPInfo" para cargar y manejar diferentes tipos de datos de manera efectiva.

Referencias:
Bryan Salazar López. (2019, June 12). Algoritmo de Dijkstra. Ingenieria Industrial Online; Ingenieria Industrial Online. https://www.ingenieriaindustrialonline.com/investigacion-de-operaciones/algoritmo-de-dijkstra/
MBR Master Boot Record. (2023, February 22). GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/mbr-master-boot-record/
Graph Data Structure And Algorithms. (2024, January 17). GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/
Estefania Cassingena Navone. (2020, September 28). Dijkstra's Shortest Path Algorithm - A Detailed and Visual Introduction. FreeCodeCamp.org; freeCodeCamp.org. https://www.freecodecamp.org/news/dijkstras-shortest-path-algorithm-visual-introduction/#:~:text=Dijkstra's%20Algorithm%20finds%20the%20shortest,node%20and%20all%20other%20nodes

Binary Heap. (2014, November). GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/binary-heap/
¿Qué es un algoritmo de ordenamiento? (2023). Platzi. https://platzi.com/tutoriales/1469-algoritmos-practico/6031-que-es-un-algoritmo-de-ordenamiento/
Guillermo, J. (2018, June 23). ¿Qué es la complejidad algorítmica y con qué se come? Medium; Medium. https://medium.com/@joseguillermo_/qu%C3%A9-es-la-complejidad-algor%C3%ADtmica-y-con-qu%C3%A9-se-come-2638e7fd9e8c