Tecnológico de Monterrey

Reflexión personal

Javier Barron Vargas
A00842507

En esta actividad no solo trato de ordenar registros, sino de construir un sistema completo para manejar una bitácora real con tiempos, fechas etc. Desde la lectura del archivo hasta la generación de los archivos de salida, cada decisión de diseño tuvo un impacto directo en el desempeño del programa, sobre todo al trabajar con muchos registros.

Uno de los puntos más importantes fue la elección de la estructura de datos. La bitácora es un conjunto dinámico de registros que debe recorrer varias veces, dividirse durante el ordenamiento y luego volverse a combinar. Por eso, usar una Doubly Linked List tuvo mucho más sentido que usar un arreglo o una Linkedlist simple. Poder recorrer la lista en ambas direcciones y manipular sublistas facilitó bastante la implementación de Mergesort y el manejo general de los datos.

Durante la carga del archivo, insertar cada registro al final de la lista fue O(1), lo cual permitió leer la bitácora de forma eficiente. Sin embargo, buscar directamente dentro de la lista siempre es O(n), así que quedó claro que ordenar previamente la información no era opcional. Solo así fue posible aplicar la búsqueda binaria para encontrar las fechas de inicio y fin del rango solicitado por el usuario sin tener que recorrer toda la lista.

La comparación entre Mergesort y QuickSort fue de lo más interesante del proyecto. Aunque en teoría ambos son O(n log n) en promedio, en la práctica su comportamiento fue distinto sobre la Doubly Linked List. Merge Sort mantuvo un desempeño estable en todos los casos, mientras que QuickSort fue más variable y en algunos escenarios se acercó a su peor caso de $O(n^2)$, especialmente cuando la lista ya estaba parcialmente ordenada. Esto confirmó que Merge Sort se adapta mejor a listas enlazadas y que QuickSort, aunque muy rápido en arreglos, pierde muchas de sus ventajas en este tipo de estructuras.

En conclusión, la actividad me ayudó a entender mejor cómo las decisiones de diseño afectan directamente la eficiencia de un sistema y no solo su correcto funcionamiento. Más allá de pasar un ejercicio, siento que este tipo de trabajos será muy útil en mi carrera de ITD, ya que en proyectos reales no basta con que un programa funcione, sino que debe ser eficiente, escalable y capaz de manejar grandes volúmenes de datos sin perder desempeño para que trabaje lo mejor posible para una empresa. Al final, este proyecto me dejó claro que la eficiencia no depende de una sola decisión, sino de cómo se combinan la estructura de datos, el algoritmo de ordenamiento y el tipo de problema, y que la elección de la Doubly Linked List junto con Mergesort no fue arbitraria, sino el resultado de ver cómo se comportan realmente estas herramientas cuando se usan para resolver un problema concreto como el manejo de una bitácora.