Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey

Programación de estructuras de datos y algoritmos fundamentales
Grupo 850

Después de clase | Tarea individual: Act-Integradora-1 Conceptos básicos y algoritmos fundamentales

Docente: Dr. Eduardo Arturo Rodríguez Tello

Jesús Alonso Galaz Reyes (A00832930)

28 de marzo de 2023

Reflexión

La importancia de utilizar algoritmos de ordenamiento y búsqueda eficientes en una situación como la presentada en el código base radica en el manejo y procesamiento de grandes volúmenes de datos, como en este caso, registros de bitácora. En el código base, se utiliza el algoritmo 'std::sort', que es una implementación del algoritmo de ordenamiento rápido o QuickSort. Este algoritmo es altamente eficiente en situaciones donde se manejan grandes cantidades de datos, ya que su complejidad temporal promedio es de O(n log n) (Cormen, Leiserson, Rivest, and Stein, 2009).

En comparación con otros algoritmos de ordenamiento, como Bubble Sort o Insertion Sort, QuickSort es considerablemente más rápido y eficiente, especialmente cuando se manejan conjuntos de datos más grandes. Por ejemplo, el algoritmo Bubble Sort tiene una complejidad temporal de O(n^2), lo que significa que el tiempo de procesamiento aumenta drásticamente a medida que crece el tamaño del conjunto de datos (Knuth, 1997). Por otro lado, el algoritmo Insertion Sort también tiene una complejidad temporal de O(n^2), pero puede ser más eficiente que Bubble Sort en ciertos casos, como cuando los datos están parcialmente ordenados (Sedgewick & Wayne, 2011).

En este problema específico, se manejan registros de bitácora que pueden contener miles o incluso millones de entradas. Utilizar un algoritmo de ordenamiento eficiente, como QuickSort, permite procesar y analizar estos registros de manera más rápida y precisa. Además, al combinar un algoritmo de ordenamiento eficiente con un algoritmo de búsqueda eficiente (por ejemplo, Binary Search), se puede agilizar aún más el proceso de análisis y selección de registros dentro de un rango de tiempo específico.

En pocas palabras, es crucial utilizar algoritmos de ordenamiento y búsqueda eficientes al abordar problemas que involucren grandes conjuntos de datos, como en este caso de registros de bitácora. Al elegir algoritmos eficientes, como QuickSort y Binary Search, se puede minimizar el tiempo de procesamiento y mejorar la precisión del análisis de los datos.

Bibliografía

- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3ra ed.). MIT Press.
- Knuth, D. E. (1997). The Art of Computer Programming, Volume 3: Sorting and Searching (2da ed.). Addison-Wesley.
- Sedgewick, R., & Wayne, K. (2011). Algorithms (4ta ed.). Addison-Wesley.
- GeeksforGeeks. (s. f.). QuickSort. Recuperado de https://www.geeksforgeeks.org/quick-sort/
- GeeksforGeeks. (s. f.). Binary Search. Recuperado de https://www.geeksforgeeks.org/binary-search/