David Alejandro Lozano Arreola A01722728

Programación de estructuras de datos y algoritmos fundamentales

Evidencia 1 - Reflexión

Al momento de tener un conjunto de datos de gran tamaño. Lo primero que se debe hacer como desarrollador es tratar de ordenarlo. Para ello, existen diversos algoritmos que se pueden utilizar. Sin embargo, cada uno de ellos tiene su complejidad de espacio y de tiempo. Es decir, necesitan memoria de la computadora para ejecutarse, y además necesitan realizar una cierta cantidad de operaciones antes de completarse.

Durante el periodo se usaron dos algoritmos de búsqueda. Linear search, el cual tiene una complejidad de O(n), es decir, en el peor de los casos tarda “n” operaciones en encontrar un valor de una lista de “n” elementos. Por otra parte, binary search tiene complejidad de O(log(n)), lo cual, nos indica que está dividiendo el conjunto de datos por la mitad a cada uno de sus pasos. Esto lo hace debido a que en una lista ordenada, es posible descartar los valores más pequeños o más grandes que un valor medio. No obstante, para poder ejecutar binary search, es necesario ordenar la lista a través de algoritmos. (Geeks4Geeks, 2017) Finalmente, en los algoritmos de ordenamiento se vieron algunos de complejidad O(n^2), los cuales necesitan recorrer el arreglo entero en cada una de sus iteraciones, por ejemplo, insertion sort necesita comparar todos los elementos hasta encontrar la posición correcta de un valor. Por otra parte, algoritmos como merge sort, tienen una complejidad de O(nlog(n)), debido a que dividen el conjunto en cada iteración.

Sin embargo, no siempre es conveniente usar el algoritmo más eficiente. Existen casos en los que otros factores como el tiempo de desarrollo o la reducción de errores son más importantes que la complejidad del algoritmo. Por ejemplo, en el caso del código de la evidencia 1. Se podría utilizar Binary search para encontrar fechas existentes en el arreglo de datos. Sin embargo, dicho algoritmo no funciona en caso de buscar fechas inexistentes. Por lo cual, en este caso es mejor utilizar Linear search.

Referencias

Binary Search Data Structure and Algorithm Tutorials. (2014, January 28). GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/binary-search/

Linear Search Algorithm Data Structure and Algorithms Tutorials. (2016, October 20). GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/linear-search/

Sorting Algorithms. (2017, June 6). GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/sorting-algorithms/