Act 2.3. Actividad Integral Estructura de datos lineales (Evidencia Competencia)

Alain Vicencio Arizabalo A01620758

Julian Andres Gonzalez Arguello A00831514

Programación de estructura de datos y algoritmos fundamentales

Grupo 570

ITESM

Domingo 29 de enero del 2022

En este trabajo se implementaron las funciones del Merge Sort, Quick Sort y Binary Search para los resultados de una bitácora en forma de una lista doblemente enlazada.

El método Quick Sort en la lista doblemente enlazada es apuntar directamente al último nodo de la lista, marcándola como el nodo del que se harán las comparaciones, usando los apuntadores para hacer el acomodo de manera ascendente. Es muy parecido a su implementación en la actividad integradora pasada, la mayor diferencia viene siendo en el uso dinámico de localización, donde la vez pasada se usaba un vector, aquí se usan apuntadores, dando una complejidad de O(n^2). (GeeksforGeeks, 2023)

El método Merge Sort también presenta un método similar al de su implementación en la actividad pasada, donde la lista doblemente enlazada se divide a la mitad, logra esto al tener dos apuntadores, uno que se mueve a un paso de un nodo, mientras que el otro a un paso de dos nodos, y es cuando el segundo nodo llega a un valor nulo que se sabe que la lista ha acabado y se encuentra el punto medio y se divide la lista. Esta acción se repite recursivamente para ambas listas recién formadas hasta el punto en que las listas solo tengan un nodo, de ahí se acomodan y se van convergiendo hasta formar una sola lista nuevamente. Este método tiene una complejidad de O(nlogn). (GeeksforGeeks, 2023).

Finalmente, se implementó la búsqueda binaria, que también tiene una similitud a su implementación en la actividad pasada. En esta implementación, se utilizan los apuntadores del inicio y final de la lista para poder encontrar el punto medio y así comparar si el valor a buscar es mayor, menor o igual al valor en ese punto medio, y de ahí moverse y aplicar la fórmula recursivamente hasta encontrarlo. Este método tiene una complejidad de O(n).

La importancia de estos métodos se expande a comparación de cómo fue en la actividad anterior, pues si bien su misma lógica es lo que les permitía tener una alta eficacia y menor complejidad, en este caso al estar usando listas doblemente enlazadas, se cuenta con una mayor agilización de la información al poder usar apuntadores en vez de un método iterativo.

Reflexión Alain: En este trabajo pudimos aplicar nuestros conocimientos sobre cómo funcionan las listas doblemente enlazadas y cómo poder traducir métodos basados en otras estructuras de datos (como vectores) a estas listas, permitiendo una agilización con el uso de los apuntadores. Aunque, si bien esto aplica para la lógica de las funciones de la lista doblemente enlazada, la implementación práctica del caso probó ser más complicada, pues debimos comprender que la bitácora y los registros se construyen a base de la lista doblemente enlazada y de sus nodos, pudiendo conectar funciones con otras. En todo caso, fue una actividad que reforzó conocimientos y se aplicó para resolver una situación. Para el método de ordenamiento, encontramos que el método Merge resultó ser más eficaz que el Quick, siendo que el primero tuvo un total de 214,671 comparaciones, mientras que el segundo uno de 344,794, una diferencia considerable, pero no abismal como lo fue entre el Merge y el Bubble sort de la actividad pasada.

Referencias:

Sin autor. (2023). Merge Sort for Doubly Linked List. *GeeksforGeeks*. Recuperado de https://www.geeksforgeeks.org/merge-sort-for-doubly-link

Sin autor. (2023). QuickSort on Doubly Linked List. *GeeksforGeeks*. Recuperado de https://www.geeksforgeeks.org/quicksort-for-linked-list/