Instituto Tecnológico y de Estudios Superiores de Monterrey

Programación de estructuras de datos y algoritmos fundamentales (Gpo 608)

Profesor:
Eduardo López Benítez

Act 2.3 - Actividad Integral estructura de datos lineales

Reflexión

Francisco Nicolás Jervis Hidalgo | A00835131

El objetivo de esta actividad es encontrar una solución para poder guardar una bitácora en una estructura de datos, ordenarla según el valor de su ip y después imprimir varias entradas de la bitácora según límites puestos por el usuario. El método utilizado para la solución de esta actividad se basa en crear una clase bitácora la cual contiene en sus atributos todos los datos utilizados por cada bitácora como hora, ip, mes, etc. Después de crear un objeto para guardar los datos se utilizó un vector para guardar todas las entradas de la bitácora.

Para el ordenamiento se utilizó el método de merge sort utilizando el valor de la ip como referencia para la jerarquía de ordenamiento. Este método tiene una complejidad de O(nlogn) lo cual lo hace muy eficiente. Finalmente, para la búsqueda de un rango de ips, se consideró utilizar binary search para el límite bajo de busqueda y despues iterar por el vector imprimiendo cada objeto hasta encontrar un vector igual o mayor al límite alto de búsqueda ya que este método tiene una complejidad de O(nlogn). Después de más análisis del problema se encontró que esto no sería posible ya que es posible que no se exista el término del límite bajo por lo que no es posible implementar este método. La solución final utiliza búsqueda secuencial hasta encontrar un término mayor o igual al límite bajo, después sigue iterando el vector imprimiendo cada objeto hasta encontrar una ip con valor mayor a la de el límite alto. Este método tiene una complejidad de O(n).

Puede existir una mejor solución que la implementada utilizando BST, ya que la búsqueda en este tipo de estructura tiene una complejidad de O(nlogn) por lo que se podría implementar esta solución para buscar un nodo el cual se acerque a el límite bajo y después de encontrar este nodo se podría recorrer el árbol de manera inorden la cual recorre el árbol de manera creciente, y se imprimiera cada nodo encontrado hasta llegar a un nodo el cual sea mayor al límite mayor de la búsqueda. Aunque el recorrido sería de complejidad O(n), la complejidad de la búsqueda es menor que la de la solución implementada por lo que al implementar BST a la solución significa que esta es más eficiente con el beneficio añadido de que al agregar un nodo al árbol este ya es posicionado en orden con una complejidad de O(nlogn) por lo que no sería necesario ordenar la estructura.