Duración de estructuras de datos

Wilson Chavarría Miranda C22114 wilson.chavarriamiranda@ucr.ac.cr

***Resumen*—En este trabajo se implementaron las siguientes estructuras de datos; Lista simplemente enlazada, árbol de búsqueda binaria, árbol rojo y negro, tabla de dispersión, y lista doblemente enlazada. Para así poder medir el tiempo de las diferentes operaciones dentro de estas, se verá representado con gráficos y comparaciones mas adelante en el documento y se mostrarán las conclusiones obtenidas.**

Palabras clave—Lista, árbol, .

I. Introducción

En este trabajo se implementaron diferentes estructuras de datos para poder ver su duración realizando las diferentes operaciones posibles en cada una de ellas, las cuales son; lista simplemente enlazada, árbol de búsqueda binaria, árbol rojo y negro y tabla de dispersión, esta última utiliza el metodo de encadenamiento, utilizando una lista doblemente enlazada, para resolver las colisiones.

A lo largo de este documento se mostrarán gráficos y se harán comparaciones para tener una mejor visión de estas estructuras de datos.

II. Metodología

Para lograr lo propuesto se implementaron las estructuras antes mencionadas en el lenguaje de programación C++, y se corrieron sus funciones con dos casos diferentes, 1000000 de números enteros aleatorios, y 100000 de números enteros ordenados.

El código se muestra en los apéndices. Este código está basado en el pseudocódigo del libro de Cormen y colaboradores [1].

Los códigos se corrieron al menos 3 veces para poder hacer los promedios de tiempo, esto se verá reflejado en el cuadro de comparación, y el promedio se verá reflejado en los gráficos presentados mas adelante en este documento.

Cuadro I Tiempo de ejecución de las estructuras

(En todos los casos, k es 1000000 = A)

		Tiempo (ms)			
		Aleatorio		***Secuencial***	
Estructura	k	Búsqueda	Eliminación	Búsqueda	Eliminación
Lista enlazada	A	75500	33000	27800	34400
Árbol búsqueda binaria	A	8	8	730000	600
Árbol rojo y negro	A	8	8	6	6
Tabla de dispersión	A	3	3	3	3

Aquí *podemos ver los tiempos en ms de búsqueda y eliminación de las estructuras mostradas en la parte de la izquierda, todas con 1000000 de elementos*,

III. Resultados

Los tiempos de ejecución de las corridas de las estructuras con los números aleatorios y ordenados secuencialmentese muestran en el cuadro I.

En general el código se ejecuta bastante rápido, por eso se pensó en una tabla en ms, sin embargo, a la hora de realizar las pruebas, se puede ver como hay un salto considerable en la búsqueda con los elementos ordenados secuencialmente en el árbol de búsqueda binaria, lo cual, hace que el resto de tiempos sean casi imperceptibles.

Los tiempos promedio se muestran gráficamente en la figura 1.

La forma de las curvas no fueron las esperadas, principalmente por lo mencionado anteriormente con el árbol de búsqueda binaria, pero también se aprecia que la lista simplemente enlazada es menos eficiente en todo, y no solo en una categoría, como lo es el arbol de búsqueda binaria.

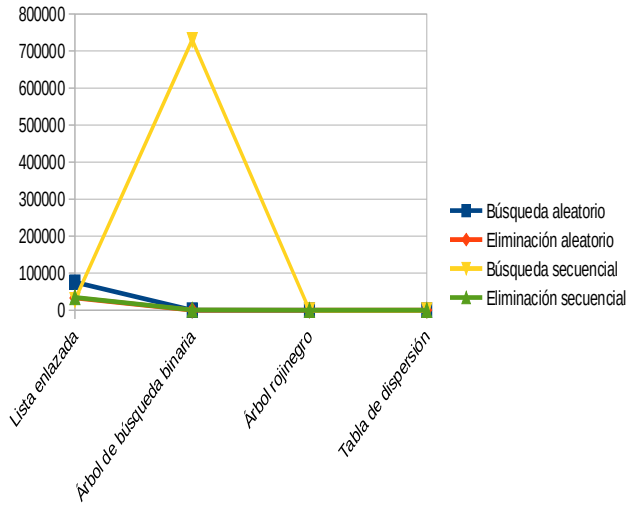

Figura 1 Gráfico comparativo de los tiempos promedio de ejecución de las estructuras de datos.

IV. Discusión

Como primer punto, con base en los resultados presentados, se puede observar que el árbol de búsqueda binaria es mucho mas lento que el resto a la hora de hacer búsqueda cuando los elementos se insertan ordenados, esto debido a que está totalmente desbalanceado, todos los elementos están en una sola rama y esto hace que por cada búsqueda tenga que bajar toda la rama hasta encontrar el elemento.

Adicionalmente, los experimentos realizados revelan que para la eliminación, la lista simplemente enlazada es la mas lenta, para ambos casos, elementos insertados en orden y aleatoriamente lo cual tiene sentido, ya que esta no presenta ningún tipo de balance, y siempre va a tratar de recorrer toda la lista para buscar el elemento a eliminar.

Por otra parte, con los datos recolectados en la tabla podemos ver que en la búsqueda, sin tomar en cuenta el árbol binario de búsqueda la lista simplemente enlazada es la mas lenta esto debido a lo mismo que en la eliminación, y el árbol es mas lento si está desbalanceado, pero esto sería un mal uso de esa estructura.

Aparte de lo mencionado anteriormente, el resto de estructuras se comportan muy similar.

V. Conclusiones

A partir de los resultados se puede concluir que la lista simplemente enlazada es la mas lenta, anque el árbol de búsqueda binaria refleje un pico en la búsqueda con los elementos insertados en orden, como se mencionó anteriormente esto es un mal uso de la estructura, y en la lista simplemente enlazada no se está haciendo un mal uso.

Al final, todas las estructuras son válidas, y dependerá de la aplicación la estructura que se quiera utilizar, para así no generar un mal uso.

Referencias

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*

Wilson Chavarría Miranda Estudiante de computación de la Universidad de Costa Rica, interesado principalmente en la parte de infraestructura.