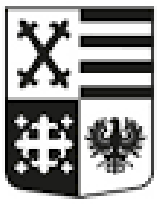

UNIVERSIDAD DIEGO PORTALES

ESCUELA DE INGENIERÍA EN INFORMÁTICA Y TELECOMUNICACIONES

FACULTAD DE INGENIERÍA Y CIENCIAS

TAREA I: Lista de Contactos

Estructura de Datos Abstractos, Lista, ABB, AVL, Hash

2018

Cristobal Nuñez[1]
1: Ingeniería Civil en Informática y Telecomunicaciones.
cristobal.nunez@mail.udp.cl

Profesor: Nicolas Rosso Chamorro.
Ayudante: Jonh Bidwell Boitano

Índice

Índice de figuras

Resumen

El siguiente informe contendrá un proyecto realizado con el objetivo de estructurar información de una lista de contactos aplicando conocimiento computaciones de datos abstractos. Se realizó repositorio en la aplicación GitHub que contendrá todas las estructuras realizadas en el proyecto, y el informe.
El objetivo de este informe es comparar todas las estructuras de datos con sus tiempos de ejecución explicando y detallando claramente la manera en que como los Contactos son agegados, buscados y eliminados.

1. Introducción

Se crearon las siguientes estructuras de datos: Lista enlazada, Arbol Binario, AVL, Arbol 2-3 y Hash. Las estructuras de datos creadas a continuación son la representación de como se organiza la información y la manera de como esta es estrcutura. El lenguaje implementado para poder crear estas estructuras de datos fue el lenguaje de programación "Python", esto es dado su versatilidad y su practica sintaxis para poder crear clases y ejecutar los comandos de manera directa.
Al haber estructurado los datos, se procedio a ejecutar un tiempo de funcionamiento para los atributos, agregar que son tanto para 10, 20 , 100 y 1000 contactos, el método buscar y el método eliminar para cada una de las estructuras que luego serán comparadas en una tabla de datos.

2. Desarrollo

Cada estructura de datos que se presentará será explicada con detalle de manera que se pueda entender la menera de como se intercan los datos, que sería una lista de Contactos, donde cada contacto debe tener tanto nombre, apellido, teléfono y dirección, ademas de generar un constructor para cada una de las estructuras empleadas.

2.1. Lista de Contacto

Creamos la lista de contactos para poder calcular los tiempos de ejecución de inserción eliminación, y busqueda. A continuación mostraremos 3 tablas que explicaran el tiempo de ejecución de las estructuras para incertar, buscar y eliminar.

Cantidad	Tiempo
10	0.7131316661834717
20	0.7327344417572021
100	0.9156436920166016
1000	2.6982922554016113

Cuadro 1. cantidad de contactos v/s tiempo de ejecución: Insertar

Cantidad	Tiempo
10	0.014017343521118164
20	0.015629291534423828
100	0.015630960464477546
1000	0.016008853912353516

Cuadro 2. cantidad de contactos v/s tiempo de ejecución: Buscar

Cantidad	Tiempo
10	0
20	0
100	0
1000	0

Cuadro 3. cantidad de contactos v/s tiempo de ejecución: Eliminar

2.2. Árbol Binario de contactos

El árbol binario es una estructura de datos que se caracteriza por por tener una cabeza y aplicarse a a través de la recursividad como operación destacada en sus nodos raíz. El atributo **root** se caracteriza por tener 2 punteros, que un puntero izquierdo señalará valores menores a si mismo, y el puntero derecho que asignará valores positivos a su nodo.

Cantidad	Tiempo
10	0.6722574234008789
20	0.6410415172576904
100	0.8223698139190674
1000	2.253451347351074

Cuadro 4. cantidad de contactos v/s tiempo de ejecución: Insertar

Cantidad	Tiempo
10	0.0124534543192188
20	0.01354343426992188
100	0.0132534019899188
1000	0.016543563255188

Cuadro 5. cantidad de contactos v/s tiempo de ejecución: Buscar

Cantidad	Tiempo
10	
20	
100	
1000	

Cuadro 6. cantidad de contactos v/s tiempo de ejecución: Eliminar

2.3. Avl de Contactos

Cantidad	Tiempo
10	0.6655089855194092
20	0.6673283576965332
100	0.9077155590057373
1000	2.5583133697509766

Cuadro 7. cantidad de contactos v/s tiempo de ejecución: Insertar

Cantidad	Tiempo
10	0.016019821166992188
20	0.018030405044555664
100	0.014540672302246094
1000	0.014539718627929688

Cuadro 8. cantidad de contactos v/s tiempo de ejecución: Buscar

Cantidad	Tiempo
10	0
20	0
100	0
1000	0

Cuadro 9. cantidad de contactos v/s tiempo de ejecución: Eliminar

2.4. Árbol 2-3 de Contactos

La ventaja que tiene un arbol B-tree es que como voy a cargar toda la información en memoría, esta estructura me dá la facilidad de cargar la información y procesarla de manera más rápida cuando analisamos grandes volumenes de información.

En un orden tiene que tener una dependencia depende directamente de la cantidad mínima datos es un dato menos que los que tienen sus hijo.

Como ejemplo, el árbol de orden 5, puede tener un máximo de 4 valores por nodo. Un nodo

Cantidad	Tiempo
10	
20	
100	
1000	

Cuadro 10. cantidad de contactos v/s tiempo de ejecución: Insertar

Cantidad	Tiempo
10	
20	
100	
1000	

Cuadro 11. cantidad de contactos v/s tiempo de ejecución: Insertar

Cantidad	Tiempo
10	
20	
100	
1000	

Cuadro 12. cantidad de contactos v/s tiempo de ejecución: Insertar

2.5. Hash de Contactos

La estructura de datos de Hash tendrá una particularidad distinta, dado que para poder realizar esta estructura requerimos agregar como parámetro un valor que será el largo de la lista de hash para todo comenzar a agrupar valores con sus key.

Cantidad	Tiempo
10	0.6610491275787354
20	0.6568746566772461
100	0.8589065074920654
1000	2.279189109802246

Cuadro 13. cantidad de contactos v/s tiempo de ejecución: Insertar

Cantidad	Tiempo
10	0.1032900810241699254
20	0.10228300094604492
100	0.10027933120727539
1000	0.09226036071777344

Cuadro 14. cantidad de contactos v/s tiempo de ejecución: Eliminar

Cantidad	Tiempo
10	
20	
100	
1000	

Cuadro 15. cantidad de contactos v/s tiempo de ejecución: Buscar

3. Análisis de Resultado

Luego de obtener claramente los tiempos de ejecución de cada uno de los procesos de inserción, búsqueda y eliminación de las estructuras de datos tanto como la lista, el árbol binario, el árbol AVL y el Árbol 2-3 como el Hashing, destacamos claramente que los tiempos de ejecución para los métodos de inserción son los que llevan un tiempo de ejecución más amplio que los demás. Esta cifra se puede concretar al definir un tiempo promedio de todas las inserciones

de todas las estructuras y comprarlas entre ellas.

$$\frac{\sum_j^n tiempo_{i,j}}{n} \quad (1)$$

La ecuación anterior representar el promedio de tiempo en que tarda en insertar datos en una estructura. Esta ecuación será la misma para emplearla en todas los demás método tales son como quitar y buscar.

En los método de búsqueda de datos en las distintas estructuras de datos, menos en la lista lineal y en el arbol 2-3, se logra ver como a medida que aumentamos el número de datos en la estructura, los tiempo también aumenta, a diferencia de los demás arboles, los tiempos de ejecución no son correlativos con respecto a la el volumen de esta, dado que al no ser un árbol perfectamente balanceado, podemos decir que el retorno, y el desplazamiento de el programa dentro de la estructura es distinto dependiendo de la profundidad de la rama en cual se desplaza.

4. Conclusión

Luego de haber obtenido los tiempo promedio de todos los métodos que hay en todas las estructuras de datos, llegamos a la conclusión que el proceso que requiero intercar algún datos está seleccionado por el de agregar, dado que este método requiere ingresar un dato, dado que en su proceso de inserción las primeras operaciones consisten en buscar el dato para luego realizar el proceso de incerción, en cambio los métodos de eliminación y búsqueda, requieren menos tiempo de

5. Bibliografía y Anexos

[1] John Bidwell & Universidad Diego Portales. Estructura de Datos. Ayudantia. `https://github.com/JohnBidwellB`