

Laboratorio Nro. 4 Tablas de Hash y Árboles

John Esteban Castro Ramírez
Universidad Eafit
Medellín, Colombia
jecastror@eafit.edu.co

Carlos Gustavo Vélez Manco
Universidad Eafit
Medellín, Colombia
cgvelezm@eafit.edu.co

1)

1.1 Ejercicio colisiones entre abejas.

Este ejercicio nos pide verificar las colisiones entre abejas que se encuentran a 100 metros o menos de distancia, estos ejercicios para calcular colisiones son muy útiles en la industria de los videojuegos, con un conjunto de datos que contiene las coordenadas geodésicas se debe retornar las coordenadas de aquellas abejas que se encuentran a menos de 100 metros de alguna otra abeja.

1.2 Ejercicio directorio

Se debe desarrollar un algoritmo que permita consultar los archivos que se encuentran en una carpeta específica, por ejemplo en mi PC, en la carpeta 'Desktop' tengo carpetas como 'Documentos universidad', 'varios', 'papelera'...; lo que permite este algoritmo es que si por ejemplo deseo consultar los archivos que se encuentran en la carpeta Desktop me retorne el nombre de dichos documentos, o los que se encuentran en la carpeta con un peso menor a 100 MB y consultas así por el estilo que pueden ser bastantes útiles para encontrar documentos fácilmente.

1.3 Árbol genealógico

Lo que pide este ejercicio es crear un árbol binario que funcione como un árbol genealógico, tomando en cuenta que no se tomarán hermanas, tías, tíos.. y permita calcular quién es la abuela materna de cierta persona. Un factor a tener en cuenta es que el árbol no puede ser de búsqueda o autobalanceado ya que en el primer caso todos los hombres quedarían de una sola rama y las mujeres de la otra y en el segundo caso se afectaría el orden genealógico.

2.1 Recorrido PosOrden

Para este ejercicio se debe retornar el recorrido posorden de un árbol binario de búsqueda dado el recorrido preorden, la clave de este ejercicio es tener en cuenta que el árbol es un árbol de búsqueda ya que dado el recorrido preorden se puede construir fácilmente para luego ejecutar el recorrido posorden.

ESTRUCTURA DE DATOS 1

Código ST0245

2.2 Suma en un árbol

El objetivo de este ejercicio es que dado un comando en forma de lista, construir un árbol binario y luego retornar 'Yes' o 'No' si se puede realizar cierta suma en alguno de los recorridos del árbol desde la raíz hasta la hoja.

3) Simulacro de preguntas de sustentación de Proyectos

3.1 La estructura de datos que usamos para calcular las colisiones entre las abejas es un octree, la cual es un tipo de árbol, en donde cada nodo tiene 8 "hijos" y es el tipo de estructura de datos que se suele usar en espacios tridimensionales, son por así decirlo el equivalente a los quadtree que se usan para espacios bidimensionales; en el siguiente gráfico podemos observar este funcionamiento.

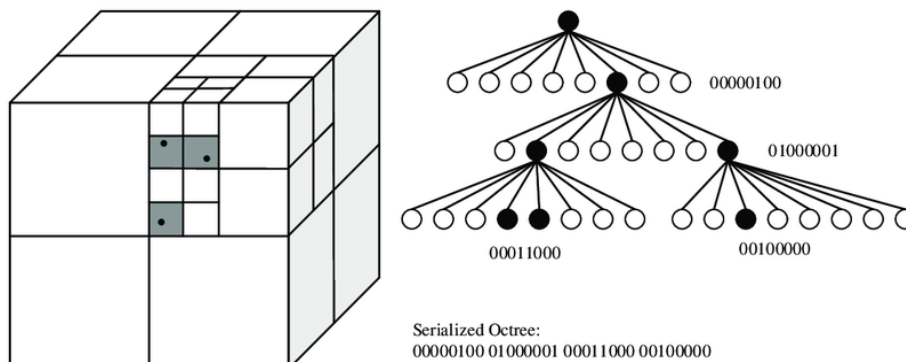


Figura 1. Ejemplo de octree, tomada de: https://www.researchgate.net/figure/Schematic-overview-of-the-octree-data-structure-and-its-serialization-Nodes-in-the-tree_fig2_330892083

Para la elección de esta estructura estábamos entre las tablas de hash espaciales y octree, ya que ambas manejan la misma complejidad, pero teniendo en cuenta que los octree suelen ser un poco más rápidos cuando el volumen de datos es muy grande en comparación a las tablas de hash espaciales, por esto hicimos la elección. Básicamente el funcionamiento del octree es localizar las abejas según su posición en el espacio (latitud, longitud, altitud) e ir ubicandolas en el octree que se compara con el cubo que vemos en la figura anterior, y las abejas las podríamos asimilar con los puntos de color negro, cuando existen dos abejas en el mismo cubo delimitado por un perímetro de 100 metros, es porque las abejas colisionarán; cabe resaltar que el algoritmo aún no está efectuado para calcular por ejemplo dos abejas que se encuentran en las esquinas de cubos distintos y que posiblemente también se puedan encontrar a menos de 100 metros.

La complejidad del algoritmo en el peor de los casos es **$O(\log n)$** , y esto se da debido a que con esta estructura no se tendrán que comparar todas las abejas con todas las abejas (lo que resultaría en $O(n^2)$), si no que se comparan con las abejas ubicadas en las celdas o cubos cercanos, lo que reduce su complejidad a logarítmica.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 1

Código ST0245

3.2 [Ejercicio opcional] No, no se puede implementar más eficientemente un árbol genealógico para que la búsqueda e inserción se puedan hacer en tiempo logarítmico; ya que, la única posibilidad para que su complejidad se reduzca es que el árbol sea autobalanceado y por lo general, para poder mantener la equitatividad de ambas ramas (izquierda y derecha) de este tipo de árbol, se suelen hacer rotaciones y esto afectaría el orden genealógico, pues posiblemente un par de personas que eran pareja, podría quedar como que uno es el padre del otro, o un hijo es el padre del padre y cosas similares que no tendrían sentido en la construcción del árbol genealógico.

3.3 [Ejercicio opcional]

Explicación de la implementación de los ejercicios 2.X

-Ejercicio 2.1. Recorrido posorden:

Para implementar este ejercicio lo que realizamos fue, teniendo en cuenta un detalle importante, el árbol era binario de búsqueda; pedir la cantidad de nodos que iba a tener el árbol que deseaba construir el usuario y luego desarrollar un ciclo en donde se pedía el número o dato que iba a tener, es decir, el recorrido preorden y luego de esto, como el árbol era binario se iba insertando al árbol cada número e iba a quedar construido correctamente ya que el recorrido era preorden(raíz, izquierda, derecha) y por último simplemente realizamos el método para imprimir el árbol en un recorrido posorden(izquierda, derecha, raíz).

3.4 Cálculo de complejidad de los ejercicios 1.X

-Ejercicio 2.1. Recorrido posorden

Para calcular la complejidad de este código ejecutamos la siguiente serie de pasos:

- 1- $T(n) = C1 + C2 + C3 * n + C4 * n + C5 * n^2 + C6 + C7 + C8 * n$ -->(Ecuación de recurrencia)
- 2- $T(n)$ es $O(C1 + C2 + C3 * n + C4 * n + C5 * n^2 + C6 + C7 + C8 * n)$ -->(Notación O)
- 3- $O(C1 + C2 + C3 * n + C4 * n + C5 * n^2 + C6 + C7 + C8 * n) = O((n * (C3 + C4 + C8)) + (n^2 * C5) + C')$
-->(Regla de la suma y factor común)
- 4- $O((n * (C3 + C4 + C8)) + (n^2 * C5) + C') = O(n * C + n^2 * C)$ -->(Regla de la suma)
- 5- $O(n * C + n^2 * C) = O(n + n^2)$ --->(Regla del producto)
- 6- $O(n + n^2) = O(n^2)$ --->(Regla de la suma)
- 7- $T(n)$ es $O(n^2)$

Así, la complejidad de este algoritmo para el peor de los casos es $O(n^2)$, en donde n es la cantidad de nodos del árbol.

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 1
Código ST0245

3.5

-Ejercicio 2.1: 'n' es la cantidad de nodos del árbol

4) Simulacro de Parcial**4.1**

- 1 **B)** Que inician con la misma letra colisionan
- 2 **D)** $O(1)$

4.2

- 1 Retorna el nodo ancestro común más cercano entre n_1 y n_2
- 2 $T(n/2)+T(n/2)+C$, que es $O(n)$
- 3 Se puede mejorar usando una condición, si n es menor que el dato entonces se va a la izquierda; en caso contrario se va a la derecha, sin ejecutar las dos recursiones y su complejidad se reduciría a $O(\log_2 n)$

4.3

- 1 return True
- 2 $O(n+m)$ pero como $n=m$ puede ser $O(n)$ ó $O(m)$

4.4

- 1 **C)** $T(n)=2T(n/2)+C$, que es $O(n)$
- 2 **A)** $O(n)$
- 3 **D)** Wilkenson, Joaquina, Eustaquia, Florinda, Eustaquio, Jovín, Sufranio, Piolina, Wilberta, Piolín, Usnavy
- 4 **A)** Cambiar el orden de las líneas 03, 04 y 05 por 05, 04 y 03

4.5

- A)** $tolInsert == p.data$
- B)** $tolInsert > p.data$

4.6

- 1 **D)** 4
- 2 return 0
- 3 if($raiz.hijos.size() == 0$)

4.7

- 1 **A)** 0,2,1,7,5,10,13,11,9,4
- 2 **B)** 2
- 3 **D)** $O(n)$

4.8

- 1 **B)** 2

4.9

- 1 **A)** 5,3,6,1,7,4,8,0,2

4.10

- 1 **B)** 2,3,4,0,5,7,6
- 2 **A)** 5
- 3 **B)** No

4.11

- 1 e.id
- 2 **A)** $T(n)=T(n-1)+C$, que es $O(n)$

4.12

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473





ESTRUCTURA DE DATOS 1

Código ST0245

























- 1 i) A=1, B=2, C=3, D=4, E=5, F=6, G=7, H=8, I=9, J=10
 2 A) G,D,B,A,C,E,F,I,H,J
 3 A) $O(n)$

6) Trabajo en Equipo y Progreso Gradual (Opcional)

6.1 Actas de reunión

	Carlos Gustavo Vélez Manco	 Saliente	2 h 22 min	24/4 17:37 ...
	Carlos Gustavo Vélez Manco	 Llamada perdida		24/4 17:36 ...

6.2 El reporte de cambios en el código

Commits on Apr 27, 2021				
Add files via upload	 johnesteban committed 2 days ago	Verified	 7a85d6c	
Delete hola	 johnesteban committed 2 days ago	Verified	 8b026e3	
Add files via upload	 johnesteban committed 2 days ago	Verified	 43dd393	
Add files via upload	 johnesteban committed 2 days ago	Verified	 ca7fb88	
Commits on Apr 26, 2021				
Delete archivo.txt	 johnesteban committed 3 days ago	Verified	 4ec0797	
Add files via upload	 johnesteban committed 3 days ago	Verified	 8b9bc85	
Delete archivo.txt	 johnesteban committed 3 days ago	Verified	 4d2c9d1	
Create hola	 johnesteban committed 3 days ago	Verified	 61d7f9c	

6.3 El reporte de cambios del informe de laboratorio

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473

ESTRUCTURA DE DATOS 1

Código ST0245

Historial de versiones	
Mostrar solo las versiones con nombre	<input type="checkbox"/>
HOY	
▶ 29 de abril, 14:04	⋮
Versión actual	
● John Esteban Castro	
29 de abril, 12:21	
● John Esteban Castro	
LUNES	
▶ 26 de abril, 18:15	
● John Esteban Castro	
SÁBADO	
▶ 24 de abril, 21:43	
● John Esteban Castro	
ESTE MES	

PhD. Mauricio Toro Bermúdez

Docente | Escuela de Ingeniería | Informática y Sistemas
 Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
 Tel: (+57) (4) 261 95 00 Ext. 9473