Laboratorio Nro. 4 Tablas de Hash y Árboles

Alejandra Palacio Jaramillo

Universidad Eafit Medellín, Colombia apalacioi@eafit.edu.co Valentina Moreno Ramírez

Universidad Eafit Medellín, Colombia vmorenor@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1 Estructura de datos implementada: árbol n-ario.

Para solucionar este problema, se diseñó un algoritmo que simula un Octree ya que, como su nombre lo indica, los 8 subnodos que tiene representan cuadrantes de las dimensiones en el espacio. Para representar este árbol, se utilizaron como estructuras base un vector dinámico y listas enlazadas ya que, en el caso de los vectores dinámicos, la complejidad para acceder a un elemento es de O(1) y, en el caso de las listas enlazadas, la complejidad para insertar. Cada casilla de este vector dinámico representa un sector o cuadrante del espacio y dentro de cada una de ellas se tiene una lista enlazada que contiene las abejas que se encuentran en ese sector. Ahora, si la diagonal en cada sector es mayor que 100, se crea otro octree dentro de este nodo y se repite el proceso recursivamente. Este procedimiento se realiza con el fin de obtener como nodos terminales sectores donde la diagonal sea menor o igual a 100 para así saber cuales son los conjuntos de abejas que colisionan.

La complejidad del algoritmo es O(nlogn), donde n es el número de abejas, ya que la complejidad de crear un octree es de O(n) y, como se crea recursivamente pero cada vez con menos datos porque estos se dividen en 8 nodos, entonces la complejidad final sería O(nlogn).

3.3 Un árbol que se muestra con una organización pre-orden, es aquel en donde se identifica primero el nodo de la raíz, para después ir indicando los valores de todo el subárbol izquierdo y cuando culmine este proceso se irá por los valores del subárbol derecho. En nuestro código, implementamos las clases, Binary Tree Node y Binary Tree para crear nuestro árbol con todos los métodos que una estructura de datos debería de tener. Para llevar a cabo los métodos recursivos, en el caso del pre-orden, se recibirá primero el valor inicial de la raíz del árbol, se imprimirá el valor que reciba el método para después realizar el llamado recursivo que recorrerá todos los elementos de sub-árbol izquierdo, y después se hará el llamado

PhD. Mauricio Toro Bermúdez









nuevamente, pero en este caso recorriendo los elementos del sub-árbol derecho. Cuando se realiza el método de pos-order cambia lo que cambiará será la ubicación de la función de imprimir, la cual irá al final del método en vez del inicio. Ya así tendremos nuestra manera de calcular los árboles de manera post-ordenada y pre-ordenada fácilmente.

- **3.5** La complejidad de todos los métodos es de O(n) ya que, para imprimir en preorden y postorden se debe pasar por todos los nodos del árbol. La complejidad de cualquiera de estos dos algoritmos, en términos de T(n) es 2T(n/2)+c, lo cual nos lleva a una complejidad lineal. El método buildingTree tiene una complejidad de O(nlogn) puesto que, mediante un ciclo for, se recorre el arreglo dado en preorden para luego insertar cada elemento en su posición en el árbol. Finalmente, la complejidad final del algoritmo que soluciona el problema planteado es O(n) + O(nlogn), lo cual se puede simplificar a O(nlogn).
- 3.6 La variable n será el número de nodos que contiene el árbol binario dado.

4) Simulacro de Parcial

```
4.1
  4.1.1 b
  4.1.2 d
4.2 c
4.3
  a. false
  b. return suma == a.data
  c. sumaElCamino(a.der, suma - a.data)
  d. | | sumaElCamino(a.izg, suma - a.data)
4.4
  4.4.1 c
 4.4.2 a
 4.4.3 d
 4.4.4 a
4.5
 4.5.1 tolnsert == p.data
  4.5.2 tolnsert > p.data
4.6
  4.6.1 d
 4.6.2 return 0
  4.6.3 \text{ raiz.hijos.size}() == 0
4.7
  4.7.1 a
```

PhD. Mauricio Toro Bermúdez

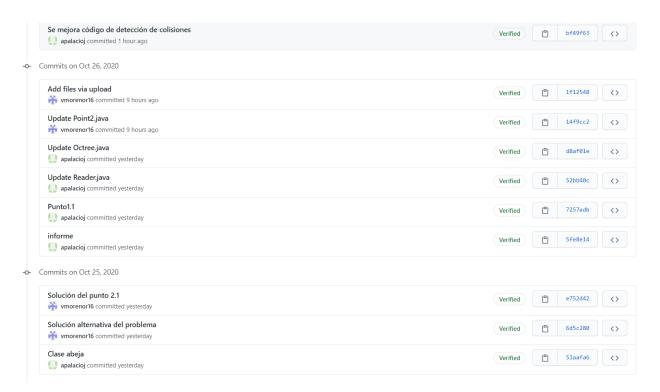






- 4.7.2 b
- 4.8 b
- 4.9 a
- 4.10 Por definir...
- 4.11
 - 4.11.1 b
 - 4.11.2 a
 - 4.11.3 b
- 4.12
 - 4.12.1 i)
 - 4.12.2 a
 - 4.12.3 a
- 4.13
 - 4.13.1 suma[e.id]
 - 4.13.2 a

5) Trabajo en equipo

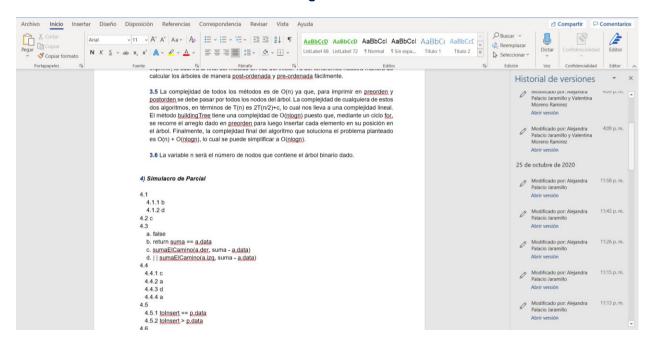


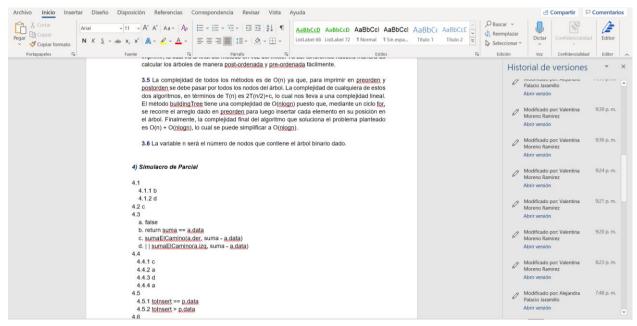
PhD. Mauricio Toro Bermúdez











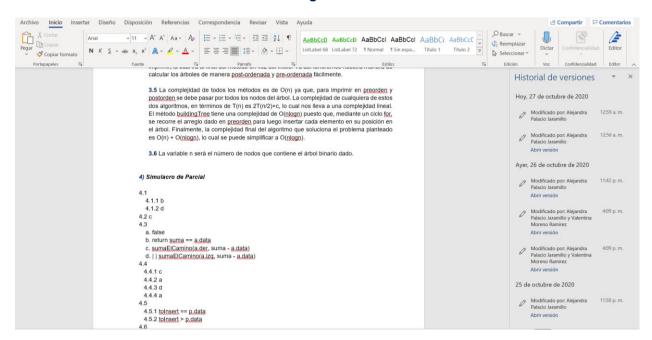
PhD. Mauricio Toro Bermúdez











PhD. Mauricio Toro Bermúdez





