Escuela Colombiana de Ingeniería Julio Garavito

Algoritmos y Estructuras de Datos 2024-1

Laboratorio Delete Binary Tree

Andersson David Sánchez Méndez

David Eduardo Salamanca Aguilar

Cristian Santiago Pedraza Rodríguez

14 de abril de 2024

DOCUMENTO TÉCNICO

Requisitos

Especificación

Completar las operaciones de un árbol binario, agregando la opción de "Delete", y que siga cumpliendo con todas las reglas AVL, teniendo en cuenta todo lo que implica eliminar un nodo. Esta acción debe considerar los siguientes aspectos: el factor de equilibrio, las rotaciones en caso de inserción o eliminación, y el balanceo.

Entrada

Las entradas varían con el método que se esté utilizando ya que se pueden insertar datos, eliminar y buscar, pero siempre se envía como parámetro un valor.

Salida

Las salidas varían de igual manera con el método que se esté utilizando ya que cuando se insertan, eliminan o buscan datos la salida es el árbol modificado con la acción correspondiente. También agregamos los métodos preOrder, inOrder, posOrder, getHeight y getAVLFactor los cuales no tienen entrada, pero si salida y es la acción correspondiente al nombre del método.

Diseño

Estrategia

Como se busca eliminar un valor del árbol, este lo interpretamos como una hoja y al eliminarlo llamamos al método correspondiente para que se balancee el árbol una vez es eliminado.

En caso de que la hoja tenga descendientes se busca el sucesor inmediatamente, si solo es uno es fácil identificarlo ya que es el propio descendiente y en caso de que sean dos se deben analizar ambas partes siempre llamando al método correspondiente que balancee el árbol.

Considerando todo lo anterior podemos dividir el funcionamiento del método delete en cuatro partes:

1. Verificar si el árbol está vacío, si este llega a estar vacío retorna None.
2. Comparación del valor a eliminar con la hoja actual.
    - Si este es menor llamamos el método delete en el subárbol izquierdo.
    - Si este es mayor llamamos el método delete en el subárbol derecho.
3. Manejo de los casos de eliminación, si el valor a eliminar es una hoja o tiene solo un descendiente, se elimina directamente esta hoja, en algunos casos retornando None y si el valor a eliminar es una hoja con dos descendientes se busca el sucesor, que es el nodo más pequeño del subárbol derecho remplazando directamente la hoja a eliminar.

4. Reequilibrio del Árbol, después de eliminar llamamos al método detectAndBalance para verificar el factor de equilibrio AVL y reestructurarlo en caso de ser necesario.

Casos de prueba

Los casos de prueba se evidencian en la ejecución del código.

Análisis

Temporal

La complejidad del método delete en el peor de los casos es de O(n) donde n es el numero de hojas del árbol (quiere decir que tendría que recorrer cada subárbol hasta encontrar el valor que va a eliminar); y para el mejor de los casos es cuando el árbol está balanceado, es decir, cuando la altura de cada árbol está dado por O(logn). Así, se puede generalizar que un árbol binario AVL, la altura debe estar entre O(logn) <= h <= O(n).

Código

```python
def delete(self, v):
    if not self.getRoot():
        return None

    if v < self.getRoot():
        self.setLeft(self.getLeft().delete(v))
    elif v > self.getRoot():
        self.setRight(self.getRight().delete(v))
    else:
        if self.isLeaf() or (self.getLeft() is None and self.getRight() is None):  # Caso en el que el nodo es una hoja o tiene un solo hijo
            return None
        elif self.getLeft() and self.getRight():  # Caso en el que el nodo tiene dos hijos
            successor = self.getRight().getMinimum()  # Encontrar el sucesor en orden en el subárbol derecho
            self.setRoot(successor.getRoot())  # Copiar el valor del sucesor al nodo actual
            self.setRight(self.getRight().delete(successor.getRoot()))  # Eliminar el sucesor del subárbol derecho
        else:  # Caso en el que el nodo tiene un hijo
            return self.getLeft() if self.getLeft() else self.getRight()

    self.detectAndBalance()  # Rebalancear el árbol después de eliminar un nodo
    return self
```

Documentación

Dentro del código.

Fuentes

Método insertado en el código compartido por el profesor sobre Arboles Binarios AVL.