AED Tema 4. Árboles de busca equilibrados (AVL)

Martín González Dios

Con árboles desbalanceados se pierde la eficiencia de búsqueda, acercándose a la de una lista, es decir O(n) para la búsqueda. Sin embargo, con árboles equilibrados se consigue una eficiencia de O(log(n)) en la búsqueda, inserción o eliminación.

Aquellos árboles que se mantienen siempre equilibrados debido a constantes reordenaciones se conocen como **árboles AVL** (Adelson-Velskii y Landis).

En este tipo de árboles, cada nodo cuenta con un campo llamado **factor de equilibrio**, que almacena la diferencia entre la altura del subárbol derecho y la altura del subárbol izquierdo para dicho nodo. Como se estudió en el tema 2, el árbol será equilibrado mientras el valor absoluto de todos los factores de carga sea **menor que 2**.

1. Inserción

Se pueden dar varios casos en la inserción:

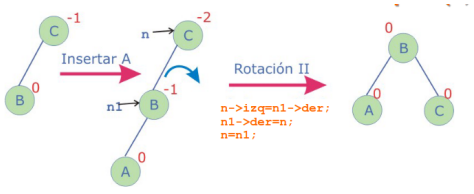
- Si las ramas izquiereda y derecha tienen la misma altura, da igual donde se inserte, que al insertar solo se producirá un nuevo factor de equilibrio de -1 o 1, con lo que el árbol sigue estando equilibrado.
- Si las ramas izquierda y derecha difieren en su altura en 1 unidad:
  - Si el factor de equilibrio era -1 y se inserta por la derecha, o si el factor de equilibrio era 1 y se inserta por la izquierda, el nuevo factor de equilibrio pasa a ser 0, por lo que se mejora el equilibrio del árbol y no hay ningún problema.
  - Si el factor de equilibrio era -1 y se inserta por la izquierda, o si el factor de equilibrio era 1 y se inserta por la derecha, el nuevo factor de equilibrio pasa a ser -2 o 2 (respectivamente), **desequilibrando el árbol**, y siendo necesaria una **reestructuración**.

2. Reestructuración

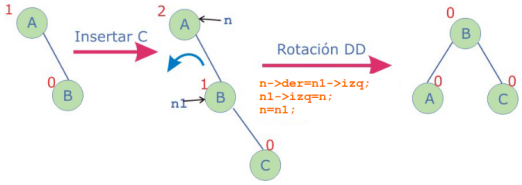
Tras una inserción, el nodo insertado pasa a ser un nodo hoja, por lo que su factor de equilibrio es 0. Posteriormente vamos subiendo por el árbol recalculando los factores de equilibrio hasta encontrar uno cuyo valor absoluto sea 2, a partir del cual habrá que reestructurar el árbol. Este proceso se repite hasta llegar al nodo raíz.

Existen 4 tipos de estructuraciones, que no son más que rotaciones del árbol: I significa izquierda y D derecha.

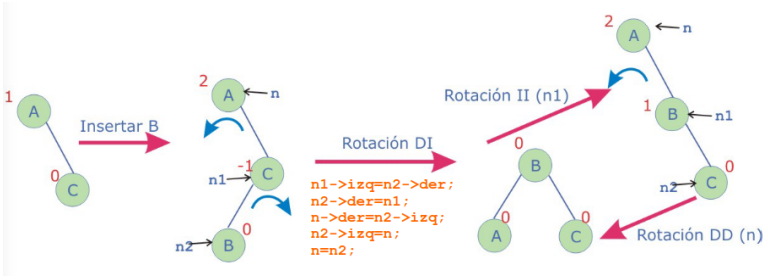
- **Rotación simple II**: se produce cuando un nodo tiene factor -2 y su hijo izquierdo, factor -1. Se rota hacia la derecha, dejando el nodo con factor -1 como nuevo padre, el nodo con factor -2 como hijo derecho, y el nodo que originó el desequilibrio, como hijo izquierdo.
- **Rotación simple DD**: se produce cuando un nodo tiene factor 2 y su hijo derecho, factor -1. Es lo contrario al caso anterior: rotación a la izquierda, nodo con factor 1 como padre, nodo con factor 2 como hijo izquierdo, y nodo que origin´o el desequilibrio como hijo derecho.
- **Rotación compuesta DI**: se produce cuando un nodo tiene factor 2 y su hijo derecho, factor -1. Se rota el hijo con factor a la derecha y se sube el nodo que originó el desequilibrio a su posición, y se tiene una rotación simple DD.
- **Rotación comuesta ID**: se produce cuando un nodo tiene factor -2 y su hijo izquierdo, factor 1. Se rota el hijo con factor 1 a la izquierda y se pone el nodo que causó el desequilibrio en su posición, teniendo un rotación simple II.



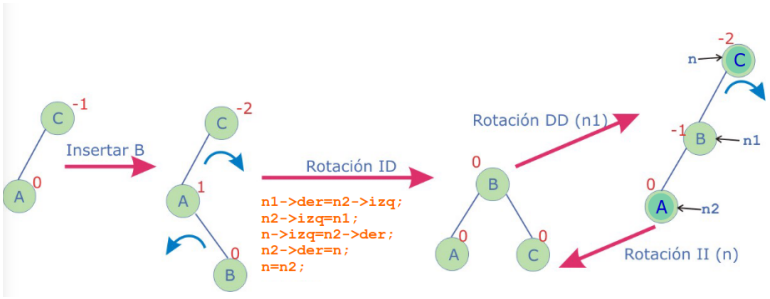
(a) Rotación simple II



(b) Rotación simple DD



(a) Rotación compuesta DI



(b) Rotación compuesta ID

3. Eliminación

Se sigue el **mismo algoritmo que para ABB**, pero **incluyendo las reestructuraciones necesarias**.
Posibles casos:

- Si el nodo a suprimir es un nodo hoja, se suprime sin más.
- Si el nodo a suprimir solo tiene un descendiente, se sustituye por su descendiente y se elimina.
- Si el nodo tiene dos subárboles, se busca el nodo más a la derecha del subárbol izquierdo, o el nodo más a la izquierda del subárbol derecho, se sustituye, se elimina, y se comprueba si es necesaria reestructuración.