1. TDA conjunto

1.1. Árbol binario AB

Un árbol binario es una estructura de datos que relaciona información de manera jerárquica no lineal, con un nodo raíz y dos subárboles disjuntos, llamados izquierdo y derecho. Cada nodo puede tener, a lo sumo, dos hijos, y el grado de un árbol binario es el mayor grado de cualquiera de sus nodos.

Algunas de las características de un árbol binario son:

- La **raíz** es el primer nodo del árbol y no tiene padre.
- Los nodos hoja son los nodos que no tienen hijos.
- Los nodos internos son los nodos que tienen al menos un hijo.
- El subárbol izquierdo de un nodo es el árbol formado por su hijo izquierdo y todos sus descendientes.
- El subárbol derecho de un nodo es el árbol formado por su hijo derecho y todos sus descendientes.
- La altura de un nodo es la longitud del camino más largo desde el nodo hasta una hoja.
- La profundidad de un nodo es la longitud del camino desde la raíz hasta el nodo.
- El nivel de un nodo es su profundidad más uno.
- La altura de un árbol es la altura de su raíz.
- El orden de un árbol binario es el número total de nodos que tiene.

Tipos de árboles binarios: Un árbol binario es un árbol en el que ningún nodo puede tener más de dos subárboles. En un árbol binario cada nodo puede tener cero, uno o dos hijos (subárboles). Se conoce el nodo de la izquierda como hijo izquierdo y el nodo de la derecha como hijo derecho.

- **Árbol binario de búsqueda:** Es un árbol binario que cumple con la propiedad de que para cada nodo, el valor de todos los nodos del subárbol izquierdo es menor o igual al valor del nodo y el valor de todos los nodos del subárbol derecho es mayor o igual al valor del nodo.

La diferencia entre un árbol binario y uno de búsqueda es que el primero no tiene ningún orden específico para organizar los elementos, mientras que el segundo sí lo tiene. Un árbol binario puede tener cualquier forma y distribución de los nodos, siempre que cada uno tenga como máximo dos hijos. Un árbol binario de búsqueda tiene una forma y distribución determinadas por la relación de orden entre los elementos.

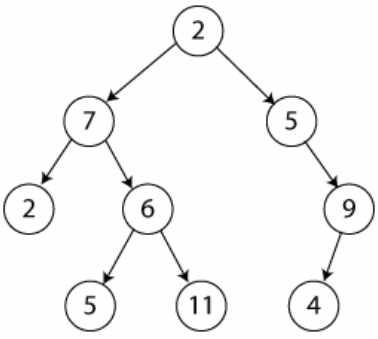

Figura 1: Árbol binario

La única condición que debe cumplir un árbol binario es que cada nodo puede tener como máximo dos hijos. No hay ninguna restricción sobre el orden de los elementos, ni sobre la forma del árbol.

1.2. Árbol binario de búsqueda ABB

Un árbol binario de búsqueda también llamado BST (acrónimo del inglés Binary Search Tree) es un tipo particular de árbol binario que presenta una estructura de datos en forma de árbol.

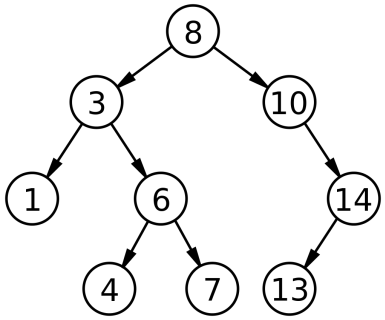

Figura 2: Árbol binario de búsqueda

1.2.1. Propiedad de ABB:

Un ABB es un AB donde (esto es lo que lo diferencia de un AB común):

- El hijo izquierdo, y todos sus hijos, son menores que la raíz.
- El hijo derecho, y todos sus hijos, son mayores que la raíz.

1.2.2. Operaciones

- **Insertar:** Inserta un elemento en el árbol.
- **Eliminar:** Elimina un elemento del árbol.
- **Buscar:** Busca un elemento en el árbol.
- **Recorridos:** Recorre el árbol en distintos órdenes.

1.2.3. Insertar

Los ABB crecen "de arriba hacia abajo", es decir que se genera un nodo cuando se encuentra nulo el puntero apropiado.

1.2.4. Eliminar o Borrado

Casos Posibles [5] :

1. **Nodo hoja:** Se elimina el nodo y se pone a nulo el puntero del padre.
2. **Nodo con un hijo:** Se elimina el nodo y se pone a nulo el puntero del padre. El hijo pasa a ser hijo del padre del nodo eliminado.
3. **Nodo con dos hijos:**

Caso 3: Eliminar nodo con dos hijos.

1. Localizar el nodo predecesor o sucesor del nodo a eliminar.
   - predecesor es "el mayor de los menores"
   - sucesor es "el menor de los mayores"
   - Para la implemtacion es igual de eficiente usar uno u otro.
2. El valor del predecesor (o sucesor) se copia en el nodo a eliminar.
3. Eliminar el nodo del predecesor (o sucesor).

1.2.5. Busqueda

Debemos comenzar por el nodo raíz e ir descendiendo a izquierda o derecha, ya sea que el valor que estemos buscando sea menor o mayor que el dato del nodo que estemos comparando

1.2.6. Recorridos

Los recorridos se clasifican en dos categorías. Video YouTube [6]:

- Profundidad: En los recorridos en profundidad se procesa cada nodo, su subárbol izquierdo y el derecho. Se usa una Pila.
  - Preorden. $\Theta(n)$
  - Inorder $\Theta(n)$
  - Postorder $\Theta(n)$
- Anchura o por niveles $\Theta(n)$: En los recorridos en anchura se procesa cada nodo por niveles. Se usa una Cola.

1.3. Arboles balanceados

Tipos de árboles balanceados:

- ABB balanceado por su altura.
- ABB balanceado por el peso.

1.3.1. ABB balanceado por su altura

Un árbol binario está balanceado por su altura con diferencia permitida d sii para todo nodo x del árbol se verifica:

$$|h_{izq}(x) - h_{der}(x)| \leq d \tag{1}$$

1.3.2. ABB balanceado por el peso

Un árbol binario está balanceado por su peso con diferencia permitida d sii para todo nodo x del árbol se verifica:

$$|peso_{izq}(x) - peso_{der}(x)| \leq d \tag{2}$$

1.4. Árboles AVL

Los árboles AVL se balancean por altura. Se trata de un tipo de árbol binario de búsqueda que cumple la propiedad de que para cada nodo, la diferencia entre la altura de su subárbol izquierdo y el de su subárbol derecho es como máximo 1.

Se agrega un atributo al nodo que es el factor de balanceo (FB). Este factor tiene tres valores permitidos: 0, 1 o -1, en cualquier otro caso se necesita rebalancear.

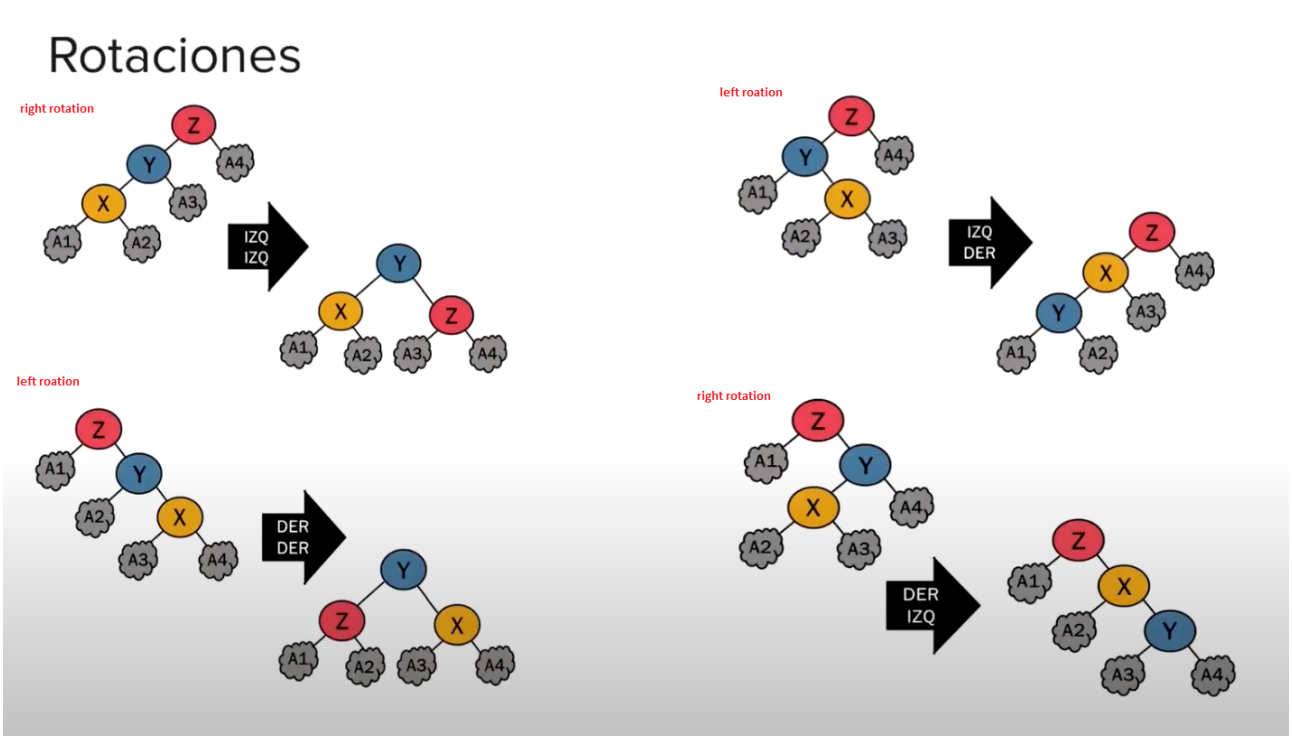

Figura 3: Rotaciones Árbol AVL

1.5. Árboles multivía o de m vías

Estas estructuras mejoran la eficiencia del almacenamiento habilitando la posibilidad de tener varios datos o claves en el mismo nodo.

Las claves siempre estarán ordenadas de forma creciente y de manera contigua, es decir que dos claves no puede haber nunca espacio no utilizado. No hay "posiciones vacías" entre dos claves.

Un árbol de m vías está formado por nodos que pueden contener hasta m-1 claves $(k_0, k_1, k_2, ..., k_{m-2})$.

Las condiciones que verifica esta estructura son:

1. Cada nodo tiene m vías (eventualmente pueden estar nulas) y m-1 lugares para almacenar claves.
2. Las claves (o datos) se almacenan de forma creciente y de manera contigua.
3. Las claves de los subárboles que están a la izquierda de la iésima clave son menores que la iesima clave.
4. Las claves de los subárboles que están a la derecha de la iésima clave son mayores que la iesima clave.
5. No esta balanceado.

1.6. Árbol B

Se trata de un árbol de m vías balanceado. Las primaras cuatro condiciones representan un árbol de m-vias.Video YouTube [7]

Las condiciones que verifica esta estructura son:

1. Cada nodo tiene m vías (eventualmente pueden estar nulas) y m-1 lugares para almacenar claves.
2. Las claves (o datos) se almacenan de forma creciente y de manera contigua.
3. Las claves de los subárboles que están a la izquierda de la iésima clave son menores que la iesima clave.
4. Las claves de los subárboles que están a la derecha de la iésima clave son mayores que la iesima clave.
5. Esta balanceado.
6. Todos los nodos excepto la raíz están completos con claves al menos hasta la mitad.

7. La raíz, o bien es hoja, o bien tiene al menos dos hijos.
8. Si un nodo tiene h claves almacenadas, entonces tiene $h + 1$ hijos (salvo la raíz y las hojas).
9. Todas las hojas están en el mismo nivel.

1.6.1. Borrado

Borrado en un árbol B:

- En hojas: normalmente.
- Nodos internos: buscar reemplazante (similar ABB).
- Puede haber underflow (nodo con menos claves de las permitidas).

Borrado con Underflow:

- **Prestar:** Si un hermano tiene más claves de las permitidas, se le puede pedir prestado.
- **Fusionar:** Si un hermano tiene la mínima cantidad de claves, se fusionan los dos nodos.

1.7. Estructura B+

Cuando es necesario acceder de manera secuencial a los datos almacenados en una estructura de arbol B, puede adicionarse una lista ligada con esos datos, la cual es accessible desde los nodos hoja.

Las condiciones que verifica esta estructura son:

- Las mismas condiciones que un árbol B.
- Todos los datos se almacenan únicamente en los nodos hoja.
- Los nodos hoja se encuentran unidos entre sí como una lista enlazada para permitir principalmente recuperación en rango mediante búsqueda secuencial.