Laboratorio I: Árboles

Estructura de Datos II

Universidad del Norte

Departamento de Ingeniería de Sistemas y Computación

Prof. Sebastian Racedo

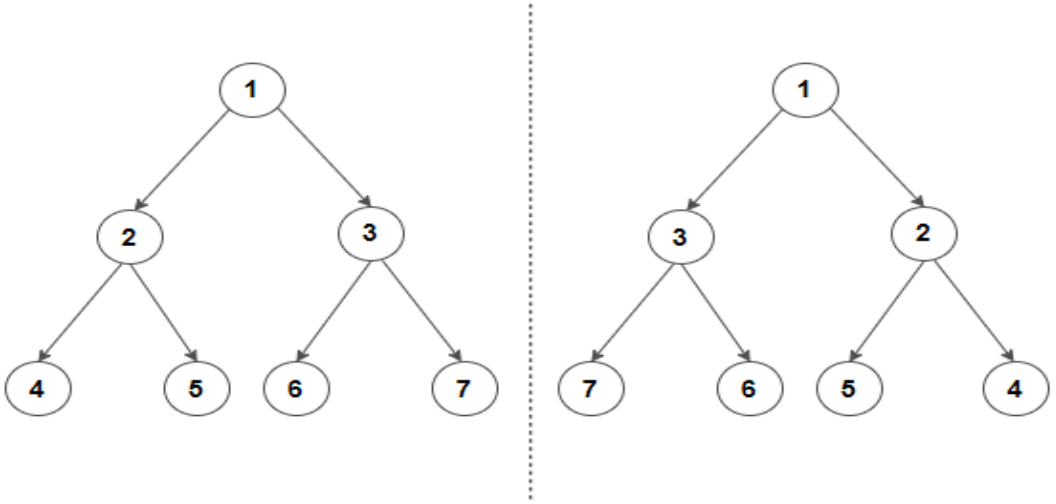
Edgar Garcia
Gabriela Bula
Lena Castillo

06.09.2022

1. (2.0) Debe construir un árbol que se autobalancee cada vez que se realice una acción, conforme lo requiera dicha acción. Concretamente este árbol debe poder realizar las siguientes operaciones:
    1. Insertar un nodo (opcional, hacer validación).
    2. Eliminar un nodo.
    3. Buscar un nodo (Queda libre como devolver si el nodo existe).
    4. Recorrido en orden por nivel (De forma recursiva).
    5. La altura dada un nodo (Recordar que la altura de un nodo es definida como el número de aristas más larga desde el nodo hasta un nodo hoja).
    6. Encontrar el abuelo o tío de un nodo dado.

    Enlace repositorio: https://github.com/EdgarGXI/Lab01_ED_II.git
2. (2.0) Investigar, plantear y resolver un problema en el que se use árboles binarios. No tiene que ser un problema de autoría propia (aunque sería bastante fructífero para ustedes). Puede ser un problema que encontraron y que les pareció de interés por sus intereses en la carrera.
    1. Investigar, plantear y definir bien un problema en el que se use árboles binarios.

    Dado un árbol binario, escriba un algoritmo que usando recursividad convierta el árbol binario en su espejo.



    2. Explicar porque este problema es de su interés.

    Nos pareció interesante realizar este problema, debido a que cuando lo piensas puedes creer que es compleja su realización, sin embargo, a la hora de la verdad es un problema que destaca por su simplicidad que consiste en usar un temporal que te permita cambiar los enlaces correspondientes a derecha e izquierda del árbol dado para convertirlo en su espejo.
    3. Resolverlo usando un lenguaje de programación. (Sin importar la autoría el código debe tener su documentación hecha por los integrantes del grupo)

3. (1.0) Investigar, plantear y resolver un ejemplo en el que se usen árboles B (o B +). El código debe estar documentado y el ejemplo debe ser de autoría propia de los integrantes del grupo.

Se tiene una lista de los datos personales de distintas personas en un documento de texto, donde cada registro corresponde a una persona distinta y cuenta cinco campos (ID, nombre, apellido, edad y ocupación), pero dado que este solo puede leerse de forma secuencial, la búsqueda de datos específicos en él no es muy eficiente. Trasladar los datos a un árbol B + no solo seguirá permitiendo hacer búsquedas secuenciales, sino que ofrecerá las ventajas de la búsqueda binaria. En el proceso de creación de este árbol, se leerá cada registro y se insertará un nodo hoja que contenga la información del registro ubicado de acuerdo a su ID. Esto a su vez puede implicar la inserción de un nodo interno con el dato del ID, así como la reestructuración del árbol.

A continuación se muestra el proceso de inserción de los siguientes registros del archivo y el árbol resultante en cada paso:

12	Chloe	Herrera	23
30	Hailey	Chapman	30
1	Nicole	Tornera	26
15	Vivian	Junco	26
20	Abraham	Hawkins	22

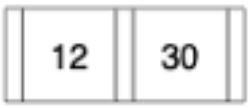
Insertando:

12	Chloe	Herrera	23

Insertando:

30	Hailey	Chapman	30

Como el ID, 30, es mayor a 12, y el árbol es de orden 3, se inserta junto al 12.

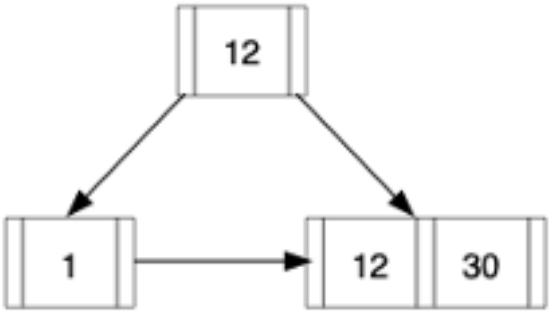


Insertando:

1	Nicole	Tornera	26

Como 1 es menor a 12, se intenta insertar el 1 a la izquierda del 12.

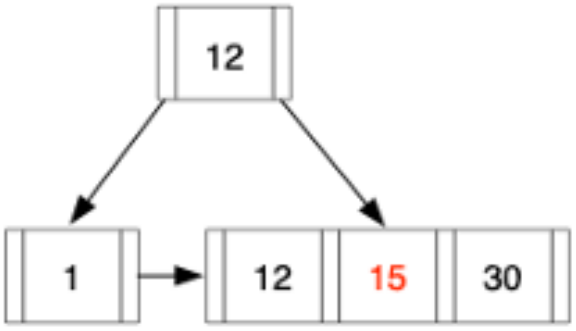
Sin embargo, siendo el árbol de orden 3, el máximo número de claves que un nodo puede contener es 2. Dado que se excedería entonces este número, la clave de la mitad de dicho nodo (12) asciende en el árbol, de forma que por su izquierda apunte hacia 1, y por su derecha apunta al nodo del que provino, donde quedó su “copia” (la cual apuntará como tal a la información que le corresponde al ID 12).



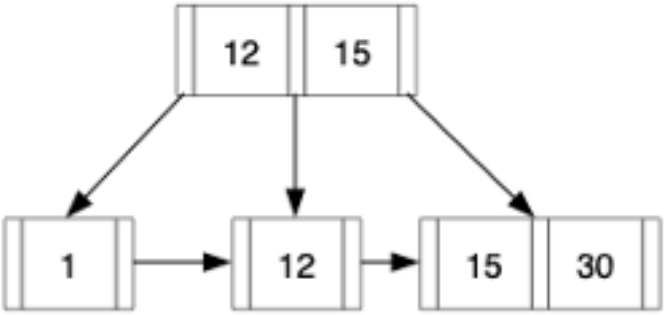
Insertando:

15 Vivian Junco 26

Al momento de insertar este nodo, inicialmente se iría entre 12 y 30. Sin embargo, esto haría que el nodo excediera el número máximo de claves.



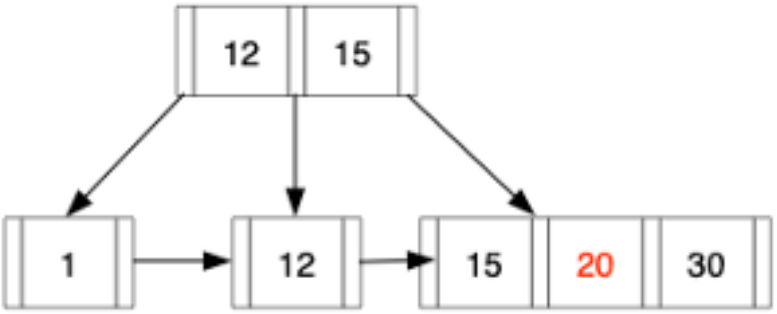
Por ello, como en el caso anterior, la clave del medio (ahora 15) asciende para situarse a la derecha de 12 en el nodo raíz, dejando una “copia” en su lugar. Así, entre las claves 12 y 15 se apuntará al nodo 12 del último nivel, mientras que el último apuntador de la raíz estará dirigido al nodo hoja con claves 15 y 30.



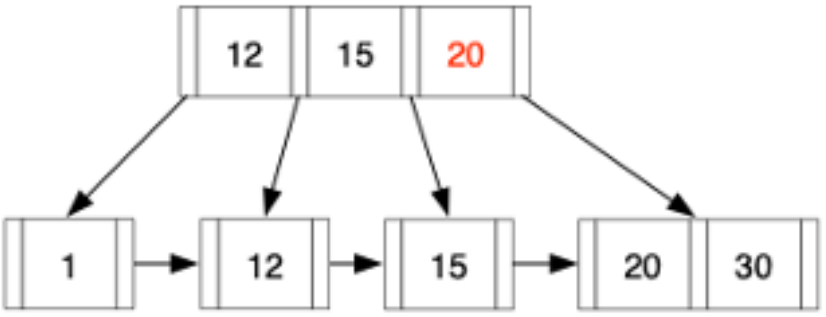
Insertando:

20 Abraham Hawkins 22

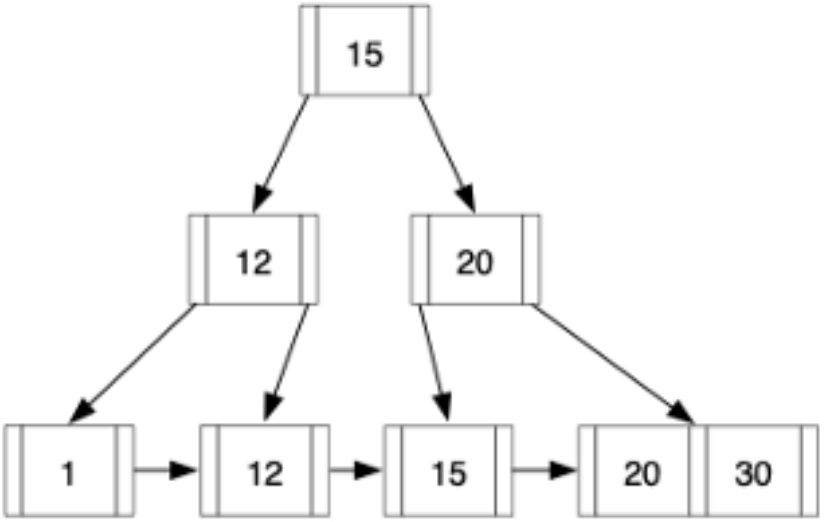
Inicialmente se desciende y se trata de insertar el 20 entre 15 y 30. Sin embargo, esto haría que se exceda el máximo de claves por nodo.



Debido a esto, 20 asciende en el árbol para quedar a la derecha de 15. Sin embargo, también en esta posición se sobrepasaría el límite de claves.



Para resolver esto, la clave del medio de dicho nodo (15) asciende en el árbol, donde queda apuntando a 12 por la izquierda y a 20 por la derecha.



Nota: El código para dar una aproximación a la solución de este punto se encuentra en la carpeta “punto_3” del repositorio. La base para el código fue tomada del artículo “Insertion in a B+ tree” de GeeksforGeeks citado en la bibliografía, el cual fue modificado para adecuarse al ejemplo.

Bibliografía

- simran_rawat, amartyaghoshgfg & aditiyadav20102001. (25 de junio, 2022). Insertion in a B+ tree. GeeksforGeeks. https://www.geeksforgeeks.org/insertion-in-a-b-tree/
- GeeksforGeeks. (17 de junio, 2022). AVL Tree | Set 2 (Deletion). https://www.geeksforgeeks.org/avl-tree-set-2-deletion/
- GeeksforGeeks. (30 de junio, 2022). Print Binary Tree in 2-Dimensions. https://www.geeksforgeeks.org/print-binary-tree-2-dimensions/