**OBSERVACIONES DEL LA PRACTICA**

Estudiante 1 Cod 202021368

Estudiante 2 Cod 201914771

# **Preguntas de análisis**

1. ¿Qué relación encuentra entre el número de elementos en el árbol y la altura del árbol?

La relación entre el número de elementos en el arbol y la altura cumple lo siguiente:

Altura ϵ[log(numero de elementos),n]

1. ¿Si tuviera que responder esa misma consulta y la información estuviera en tablas de hash y no en un BST, cree que el tiempo de respuesta sería mayor o menor? ¿Por qué?

El tiempo de respuesta sería considerablemente mayor porque las tablas de hash no tienen un método para encontrar rangos de llaves, por lo que tocaría hacer un ordenamiento en las llaves implicando una complejidad mayor. En cambio, los BST tienen operaciones efectivas para encontrar llaves en un rango.

1. ¿Qué operación del TAD se utiliza para retornar una lista con la información encontrada en un rango de fechas?

La operación del TAD que se utiliza para retornar una lista con la información encontrado en un rango de fechas es:

om.values(analyzer['dateIndex'], initialDate, finalDate)

A partir de la fecha inicial y final ingresada, devuelve las llaves del rango especificado, lo que permite retornar una lista con la informacion encontrado entre las fechas.

Esta función está estructurada recursivamente de la siguiente manera:

def values(*map*, *keylo*, *keyhi*):

    """

    Retorna todas los valores del arbol que se encuentren entre

    [keylo, keyhi]

    Args:

        map: La tabla de simbolos

        keylo: limite inferior

        keylohi: limite superiorr

    Returns:

        Las llaves en el rago especificado

    Raises:

        Exception

    """

*if* (map['type'] == 'BST'):

*return* bst.values(map, keylo, keyhi)

*else*:

*return* rbt.values(map, keylo, keyhi)

**Laboratorio 9**

**1. ¿Qué diferencia existe entre las alturas de los dos árboles (BST y RBT)?**

**BST**

Captura de pantalla de computadora

Descripción generada automáticamente

**RBT**

Captura de pantalla de computadora

Descripción generada automáticamente

Se evidencia que cuando se implementa un árbol binario de búsqueda (BST), este tiene una altura de 29, mientras que cuando se cambia por un árbol balanceado rojo-negro (RBT), esta disminuye a 13.

Así, la altura del BST es más del doble de la del RBT.

**2. ¿Por qué pasa esto?**

Esto pasa porque los RBT están balanceados (esto significa que la diferencia entre la altura del hijo izquierdo y el hijo derecho de la raíz es menor o igual a 1 y que los hijos de esos hijos están, a su vez, balanceados [noción recursiva]). Esto garantiza una repartición uniforme de los elementos. La longitud deseada de las ramas (desde la raíz hasta una hoja) es de log(N) cuando sólo se tienen arcos ólo negro. En el peor caso, la longitud será de 2log(N) [para nodos tipo 3].

Mientras que, si un árbol no está balanceado, la diferencia de altura entre el hijo derecho y el hijo izquierdo de la raíz puede ser cualquiera. Así, en el peor caso (todos los elementos terminan en un solo hijo), se genera una estructura lineal generando que la altura del árbol sea igual al número de elementos (N).

En conclusión, al tener un árbol balanceado, se garantizan ramas de igual o cercana longitud, distribuyendo los elementos de manera equitativa; mientras que si no está balanceado, la diferencia entre la longitud de ramas puede ser muy grande, llevando a que los elementos se acumulen en una rama; en el peor caso, la longitud de esta puede ser igual al número de elementos. Así mismo, la altura de un árbol se puede entender como la longitud de su rama más larga y es evidente que N>2log(N) [peor complejidad BST > peor complejidad RBT].