Laboratorio 7 EDA

Andrés Mugnier Zuluaga- 20172994

David Burgos Mendez - 201818326

- ¿Qué relación encuentra entre el número de elementos en el árbol y la altura del árbol?
  Se obtuvo el siguiente resultado:

```
Cargando información de crimenes ....
Crimenes cargados: 319073
Altura del arbol: 29
Elementos en el arbol: 1177
Menor Llave: 2015-06-15
Mayor Llave: 2018-09-03
```

La altura de un árbol determina cuantas casillas quedan libres y al momento en que se agregan los datos se van distribuyendo a lo largo de toda la estructura, es muy posible que al terminar este proceso los datos no estén repartidos uniformemente en las diferentes ramas. Por lo que la altura del árbol funciona mejor a la hora de analizar el espacio disponible y no aspectos espaciales como que rama puede contener mas o menos datos.

- ¿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?

Con los inputs 2017-01-01, 2018-01-01 se obtuvieron los siguientes resultados:

```
Buscando crimenes en un rango de fechas:
Fecha Inicial (YYYY-MM-DD): 2017-01-01
Fecha Final (YYYY-MM-DD): 2018-01-01

Total de crimenes en el rango de fechas: 101143
```

En un BST retornar los elementos que están en un rango de llaves equivale a encontrar la llave que tenga como valor el limite inferior del rango en el que se quiere buscar (en ese caso la fecha inicial) y luego, simplemente retornar el hijo derecho de este nodo.

Si el BST estuviera balanceado, buscar esta llave con el limite inferior tendría un costo de log(N) (hacer log (N) comparaciones) para luego retornar su hijo derecho.

Si no estuviera balanceado, en el peor caso una búsqueda en el BST tendría una complejidad proporcional a N+1 donde N es la altura del árbol.

Si estuviéramos en una tabla de hash, no tendríamos necesariamente un orden en los elementos. Por lo tanto, para buscar todos los elementos en un rango tendríamos que buscar fecha por fecha, introducir esta fecha en la función de hash y de compresión y luego mirar si esta llave si está en la tabla de hash para guardarla en la lista que se va a retornar. Por lo tanto, el tiempo que va a tomar esta operación en un tabla de hash va a depender del número de llaves que puedan existir dentro del rango que se desea buscar.
Por ejemplo, en este caso entre 2017-01-01 y 2018-01-01 hay 365 posibles fechas para buscar, por lo tanto, tendríamos que ver si hay crímenes en cada una de estas fechas. La operación de buscar si un elemento existe en una tabla de hash puede tardar 2.5 comparaciones (linear probing en el caso ideal) o en separate chaining puede tardar O(1) en el mejor caso y O(n) en el peor caso.

Por lo tanto, siempre va a ser mejor realizar esta operación en un BTS, pues el tiempo de ejecución no va a depender del rango de fechas a buscar, es constante para un BST fijo (log(N) para balanceado y proporcional a N si no estuviera balanceado). En este caso N=29, que es la altura del árbol.

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

  Con los inputs 2017-01-01 y Vandalism se obtuvieron los siguientes resultados:

```
Buscando crimenes x grupo de ofensa en una fecha:
Fecha (YYYY-MM-DD): 2017-01-01
Ofensa: Vandalism

Total de ofensas tipo: Vandalism en esa fecha:  12
```

  La operación del TAD orderedmap.py que retorna una lista con la información encontrada en un rango de fechas es la función *values(map, keylo, keyhi)*. Esta función toma como input el mapa ordenado en el que se quiere buscar, la llave del límite inferior que se quiere buscar y la llave del límite superior del rango que se quiere retornar. Esta función retorna solo los values de los nodos que estén en este rango como una lista.