
Campus Querétaro

Programación de estructuras de datos y algoritmos fundamentales
(Gpo 602)

# Reflexión Actividad Integral de BST (3.4)

Presenta
David René Langarica Hernández | A01708936

Profesores
Francisco Javier Navarro Barrón

17 de noviembre de 2022

En la presente evidencia implementamos un código haciendo uso de una estructura heap para llegar al mínimo número de comparaciones posibles en cada caso en particular; por ello se utilizó un min-heap para la aplicación de la fórmula para obtener dicho número de comparaciones. Esto puede dar una primera idea de las diversas aplicaciones que tiene el emplear los árboles binarios de búsqueda (BST) como estructuras de datos.

La principal ventaja de usar BST es la complejidad que estos tienen para realizar distintas operaciones como las de buscar, insertar o eliminar, dicha complejidad es de $O(log^2 N)$ lo que las hace estructuras de datos muy convenientes frente a otras de la rama cuando está ordenado y equilibrado. Lo anterior debido a que tiene la característica que se sabe desde un inicio por cuál "camino" seguir. Con la búsqueda, por ejemplo, con una comparación con la raíz se puede saber por cuál rama proseguir, si es menor por la izquierda y si es mayor por la derecha.

La característica jerárquica mencionada anteriormente tiene un sinfín de aplicaciones. Por ejemplo, una base de datos para la nómina en una empresa, en donde la raíz es el CEO y los empleados se irán acomodando de acuerdo a si tienen menor o mayor jerarquía que el CEO y así sucesivamente con los siguientes listados en la nómina.

Otra ventaja del BST es que estos sistemas permiten la concurrencia de operaciones, por lo que puede haber varias transacciones en curso a la vez. Esto es útil cuando se tienen grandes volúmenes de datos, ya que al momento de realizar una operación (como una búsqueda, insertar o eliminar, reiterando) el usuario no tiene que esperar a que su trabajo se complete para poder realizar otro.

Por otra parte, estos tienen un uso aún más complejo con el algoritmo de compresión o encriptación de datos de Huffman, el cual tiene un estilo similar al abarcado en esta evidencia, pues los BST son usados en conjunto con la estructura heap.

Gutiérrez (2017) menciona que este algoritmo "se usa para la creación de códigos de Huffman, desarrollado por David A. Huffman. Es una técnica para la compresión de datos, en otras palabras, busca reducir la cantidad de datos o espacio de memoria para crear un mensaje". Del mismo modo, Gutiérrez (2017) también menciona que "En este algoritmo es necesario el uso de un alfabeto con n caracteres finitos. Por la tanto, debes tener en cuenta que tipo de símbolos se podrán tener".

Para el algoritmo de Huffman es de suma importancia el emplear BST, ya que este algoritmo puede llegar a tener una gran cantidad de datos con el cual cada letra de un mensaje tendrá un valor a proporcionar en la creación de un nuevo mensaje comprimo, y con el que otras estructuras podrían llegar a tardarse más tiempo (y ocupar más memoria), a razón de que la encriptación puede llegar a ser uno de los algoritmos más sofisticados en ciertos casos.

Por lo cual es preciso decir que los BST tienen un gran importancia y eficiencia para problemas del tipo de los abarcados en el presente, es por ello que es una solución mucho más eficiente al tratar con grandes volúmenes de información, además de ser un sistema muy seguro dado que los datos se encuentran protegidos por la propia arquitectura del BST, y este es un dato que se puede comprobar con el estudio de distintos algoritmos para diversas aplicaciones.

## Referencias

Gutiérrez, P. (2017, 4 septiembre). *Algoritmo de Huffman - BackStreetCode*. Medium.

https://medium.com/@aprendizaje.maq/algoritmo-de-huffman-8523c21a1b1a