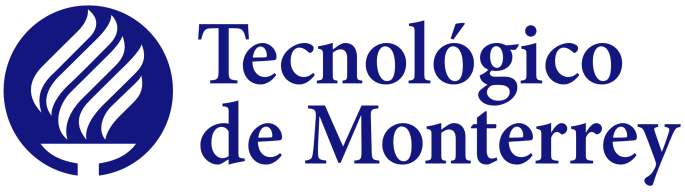
**Instituto Tecnológico y de Estudios Superiores de Monterrey**
Campus Guadalajara




**Programming of Data Structures and Fundamental Algorithms**

# Act 3.4 - Comprehensive BST activity

**Santiago Vera Espinoza**
A01641585

TC1031, Grupo 613

Jorge Enrique González Zapata

Octubre 2022

**Importance of BST's in this project**

The use of a BST in this project is very important so we can optimize the search algorithm, BST's have a very fast execution process of approximately Ω(log(n)). This are function complexities and descriptions according to "GeeksforGeeks" (2022):

- Searching: For searching element 2, we have to traverse all elements (assuming we do breadth first traversal). Therefore, searching in binary trees has the worst case complexity of O(n).
- Insertion: For inserting element as left child of 2, we have to traverse all elements. Therefore, insertion in binary trees has the worst case complexity of O(n).
- Deletion: For deletion of element 2, we have to traverse all elements to find 2 (assuming we do breadth first traversal). Therefore, deletion in binary trees has the worst case complexity of O(n).

Network infection

The way of discovering if a network is infected or not is by searching for most requests from the same IP's. This gives good arguments for thinking that the network is infected because the access requests are very frequent and they could be being done by a bot or an automated process.

Filtering IP addresses in this manner allows you to monitor the conversations taking place between particular machines, so if you suspect that a computer is infected, you can take a closer look at its traffic. It's a good idea to regularly inspect hosts generating the greatest traffic volume, as this can indicate the host is infected with malware and is attempting to spread it to other machines. (Comparitech, 2022)

**Algorithm description**

- extractIP() - Complexity O(1):

Extracts the ip from the line of text found by inFileMain().

It receives a line of text and counts the spaces to extract the section of the line that includes its respective IP.

- hashIP() - Complexity O(1):

Función para obtener el hash dado una IP.

Retorna la IP vuelta un número agregándole sus respectivos ceros faltantes, ya que cada punto divide la prioridad de dicho espacio, creando un IP de notación decimal científica que representa la prioridad de forma implícita.

- inFileMain() - Complexity O(n):

Lee el archivo e inserta la ip contando sus respectivas repeticiones.

Lee el archivo y genera un contador que en cada línea aumenta si es que su respectiva ip ha sido repetida. Una vez no se repitan dichas IPs, se agrega la key al hashmap con su respectiva ip.

- findFiveLargest() = Complexity O(n):

Itera por el BST y encuentra las llaves máximas de todo el BST.

Genera un vector de cinco lugares y evalúa en orden de prioridad si un nodo iterado por el BST es mayor a cada uno, en cuyo caso se actualiza el vector.

## Referencias


Comparitech. (2022, August 22). *How to Perform a Network Virus Scan*. Comparitech. Retrieved October 26, 2022, from https://www.comparitech.com/net-admin/network-virus-scan/

GeeksforGeeks. (2022, August 1). *Complexity of different operations in Binary tree, Binary Search Tree and AVL tree*. GeeksforGeeks. Retrieved October 26, 2022, from https://www.geeksforgeeks.org/complexity-different-operations-binary-tree-binary-search-tree-avl-tree/