Instituto Tecnológico y de Estudios Superiores de Monterrey

TC1031.850

Programación de estructuras de datos y algoritmos fundamentales

Actividad

Reflexión actividad integradora 3

Estructura de datos lineales

Profesor

Eduardo Arturo Rodríguez Tello

Presenta

Arturo Azael Godinez Rodriguez | A01641179

Fecha:

14/05/2023

Arturo Godinez:

En la actividad integradora se pidió utilizar un algoritmo de ordenamiento Heap Sort que utiliza una estructura de datos llamada Heap que utiliza para ordenar un conjunto de elementos. El heap es una estructura de árbol binario completo en que el valor de cada valor nodo es mayor que el valor de sus nodos hijos o el valor de sus nodos hijos es menor.

1. Se construye un Heap a partir del conjunto de elementos que se desea ordenar.
2. Se extrae el nodo raíz (que es el mayor elemento en un Heap máximo o el menor en un Heap mínimo) y se coloca al final del arreglo.
3. Se reconstruye el Heap con los elementos restantes.
4. Se repite el proceso de extracción y reconstrucción del Heap hasta que todos los elementos hayan sido extraídos.

En si el Heap Sort es un algoritmo muy eficiente en términos de tiempo de ejecución, con una complejidad de tiempo de O(n log n), donde n es el número de elementos a ordenar. Sin embargo, tiene una complejidad espacial de O(n), lo que significa que su consumo de memoria aumenta a medida que se agregan más elementos al Heap. Por lo tanto, el Heap Sort puede ser menos eficiente que otros algoritmos de ordenamiento como el Quick Sort o el Merge Sort en términos de uso de memoria.

También se pedía implementar Binary Heap (Priority Queue) el cual es una estructura de datos que se utiliza para cola de prioridad. En esta se busca cada elemento de una esctructa de datos en la que cada elemento tiene una prioridad asociada y los elementos se procesan a la función de su prioridad. Los elementos con la prioridad mas alta se procesan primero. Dentro esta estructura existe el min-heap y el max-heap con que se pedía en la actividad se utilizó el max-heap el cual es una estructura de datos en la que el elemento con el valor más alto se encuentra en la raíz del árbol. Los elementos se almacenan en un árbol binario completo y cada nodo tiene un valor mayor o igual que los valores de sus hijos. Esta estructura se utiliza comúnmente para implementar una cola de prioridad en la que los elementos se procesan en orden descendente de acuerdo a su valor. Las operaciones de inserción y extracción tienen una complejidad de tiempo de O(log n), donde n es el número de elementos en el Heap.

El BST (Binary Search Tree) es una estructura de datos muy útil en situaciones donde es necesario buscar y ordenar elementos de manera eficiente. En el caso de la detección de

malware en una red informática, el uso de BST puede ser beneficioso para almacenar y organizar los datos de los sistemas y dispositivos conectados.

El uso de BST en la detección de malware puede mejorar la eficiencia de la búsqueda de patrones de actividad maliciosa en grandes conjuntos de datos. Sin embargo, es importante recordar que las herramientas de análisis de seguridad deben utilizarse en conjunto con otras medidas de seguridad para proporcionar una protección completa y efectiva contra las amenazas cibernéticas.

En respuesta a la pregunta ¿Cómo podrías determinar si una red está infectada o no? Podemos tener datos almacenados en el cual se puede utilizar un BST para buscar patrones específicos de actividad maliciosa, como direcciones IP sospechosas, archivos o procesos que se ejecutan en los dispositivos conectados, y otros indicadores de compromiso (IOC).

Referencias:

GeeksforGeeks. (2023). Priority Queue using Binary Heap. *GeeksforGeeks*. https://www.geeksforgeeks.org/priority-queue-using-binary-heap/

GeeksforGeeks. (2023). Heap Sort. *GeeksforGeeks*. https://www.geeksforgeeks.org/heap-sort/

Estructura de datos del «árbol» 1: Árbol de búsqueda binaria (BST) - programador clic. (s. f.). https://programmerclick.com/article/6398838241/