Juan Carlos Garfias Tovar, A01652138

Reflexión Final

A lo largo del semestre tuve la oportunidad de poder trabajar con distintas estructuras de datos, las cuales me ayudaron a comprender como funcionan, su potencial y sobre todo entender donde al igual que cuando utilizarlas. El primer entregable basado en un sistema de ordenamiento pude aprender sobre como existen diferentes algoritmos y como a la larga el tiempo de ejecución a partir de Big O notation es sumamente relevante. Al ir incrementando la cantidad de datos la eficiencia se volvía mas importante. Si bien Quicksort era un algoritmo rápido esta tenia un rendimiento peor al merge sort pero la consistencia en general del algoritmo permitía que el vector creciera. Entender este tipo de casos me ayudó a entender por que para este tipo de estructuras lineales el merge sort es ideal. Por otro lado en cuanto a la búsqueda pude comprender como funionaba y logre implementar el binary search como algoritmo principal. Sin duda esto fue clave ya que si bien el algoritmo tenia en esencia este funcionamiento pude entender que en variadas ocasiones estos se modifican para poder lograr comportamientos específicos como el de buscar en rangos o el de buscar primeras instancias en los registros. Este algoritmo demostró ser mucho mas eficiente que una búsqueda secuencial debido a como va partiendo los datos. En el segundo entregable pude aprender el uso sobre apuntadores para crear listas enlazadas, las cuales tienen ventajas frente a vectores especialmente en como se manejan los datos. Este entregable mejoro considerablemente la complejidad pasando a un O(n log n) con el merge sort pero tomando las ventajas de la lista enlazada.

El uso de heaps fue otro elemento importante para comprender la importancia de la eficiencia y buscar soluciones que sean mucho más rápidas que otras. El heap sort permite una lectura rápida siendo igualmente un O(n log n) sin embargo, este se mantiene consistente en grandes cantidades de datos y al utilizar este tipo de estructura se pudo ordenar rápidamente de manera ascendente las ips a partir de las interacciones con los registros fallidos. Siendo mucho mas eficiente que con un ordenamiento tradicional y realizando un acceso iterativo. Pasando a los grafos estos resultaron ser la estructura mas importante para comprender como los datos pueden tener muchas dimensiones y como el uso de adyacencias es clave para realizar búsquedas y ordenamientos de manera rápida y eficiente. Los grafos al funcionar con nodos permiten un ordenamiento y almacenamiento de datos de manera mucho mas directa y rápida que en estructuras lineales. Por ello este tipo de estructuras permite dar mas información, la cual es clave para el rastreo de IPs maliciosas como lo son los intentos de login y su relación con otros intentos fallidos. La ejecución de grafos resulto ser mucho menos costosa, sin embargo la estructura mas eficiente de todas fue el Hash, el cual siendo quadratic permitía a través de su función de hashing realizar el establecimiento de keys para cada uno de los elementos de la lista de adyacencia. Logrando asi acceder a los elementos de manera directa y sin requerir iteración pasando asi de algún algoritmo de búsqueda

lineal o de O(n log n) a o(1). Sin duda este método es el mas eficiente y demuestra el por que estructuras como diccionarios son tan populares tanto para almacenar datos como para encriptar con sistemas como SHA256.

Sin duda un elemento importante en todo programador es la capacidad de entender sus áreas de oportunidad y como mejorar el código de manera cíclica hasta lograr la máxima eficiencia. Sin duda los primeros entregables a pesar de tener un diseño limpio no eran muy eficientes debido a la linealidad de los datos, implementar algoritmos de ordenamiento para estructuras no lineales es el paso mas importante para poder lograr una mejor eficiencia. A su vez, considero que uno de los errores mas grandes cometidos en los entregables fue que la mayoría de los objetos eran creados durante la ejecución y no durante la compilación. Este tipo de paradigmas es clave para poder mejorar el rendimiento de cada uno de los entregables. Por otro lado, considero que el ultimo entregable podría haberse realizado de la mano de un diccionario, esto ya que el nodo pudo haber sido creado como a partir del key y el acceso al nodo hubiese sido igual de eficiente que pasar la lista de adyacencias a un Hash separado. Esto ya que como el Hash se puede acceder a los elementos a partir de la llave. Este tipo de estructuras duales son excelentes para el manejo de datos de manera eficiente y me hubiese encantado haber realizado mas ejercicios con ellas. Considero que la eficiencia general de todos los entregables hubiese sido mayor al implementar esta estructura sobre las otras. Sin embargo, es importante denotar que a pesar de que una estructura es eficiente esto no indica que se debe de utilizar para todos los casos, principalmente ya que es mucho mas sencillo estar modificando las estructuras lineales tanto en inserts como en delets. Mientras que en grafos este tipo de delets puede afectar el balance del árbol y afectar considerablemente como se comporta. Requiriendo asi de funciones secundarias que reestablezcan el balance. Gracias al curso considero haber comprendido mucho mas como se utiliza y funcionan las distintas estructuras, la importancia de cada una de ellas y comprender por que han sido creadas para asi emplearlas en mi día a día como desarrollador e ingeniero.