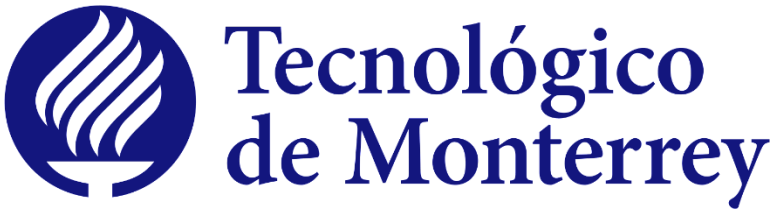
Instituto Tecnológico y de Estudios Superiores de Monterrey



Programación de estructuras de datos y algoritmos fundamentales

Tarea individual: Act-Integradora-5 - Uso de Hash Tables

Profesor: Dr. Eduardo Arturo Rodríguez Tello

Marlon Yahir Martínez Chacón - A01424875

10/06/2023

Importancia y eficiencia del uso de las tablas hash

En esta actividad integradora se nos pidió almacenar toda la información que nos estaba dando acerca de los diversos ataques que habían sucedido a todos los nodos (13370), el archivo de información consta de 13370 direcciones ip junto a la información del día y hora del ataque además de la dirección ip, la dirección ip a la que se intentó acceder, el peso de la conexión y la causa de la falla.

Como nos estaban dando una conexión entre direcciones ip que tenía un peso, lo primero que se tuvo que hacer fue utilizar la estructura de datos de la anterior actividad integradora, es decir, los grafos que usamos para poder obtener esa relación entre ip accede a ip que está accediendo. Una vez que se logró crear el grafo con la información de todos los 13370 nodos se insertaron todos los nodos en una tabla hash, esto fue porque es la estructura más eficiente para obtener la información que se nos solicitaba en esta actividad.

La información que se nos solicitaba era el número de colisiones que se genera al insertar los datos en la tabla hash cerrada, esto es debido a que el proceso para poder insertar datos es mediante una llave a diferencia de otras estructuras donde el índice que se le da al dato es el orden en el que fue insertado en la estructura, pero esto es diferente en las tablas hash porque el índice que recibe el dato que se inserto en la tabla es el resultado de hacer una operación con la llave de cada uno de los datos y el tamaño máximo de la tabla que se recomienda que sea un número primo, esta puede ser una suma, multiplicación, división, etc; por lo que varios datos pueden tener el mismo índice haciendo que haya una colisión y se tiene que mover ese dato a otra celda para que se pueda acomodar en la tabla hash, el número de colisiones en la tabla se puede disminuir al hacer más grande la tabla, usando los datos que se nos proporcionaron se obtienen los siguientes ejemplos:

Tabla hash con un tamaño de 14009

```
El tamaño maximo de la tabla hash es: 14009
Numero de colisiones: 6355
```

Se nos da un total de 6355 colisiones entre los datos

Y ahora, una tabla hash con un tamaño de 15541 elementos

```
El tamaño maximo de la tabla hash es: 15541
Numero de colisiones: 5744
```

Disminuya a 5744 colisiones

Y de esta forma podemos hacer más eficiente la colocación de datos en la tabla hash solamente aumentando el tamaño máximo de la tabla. Haciendo que nuestra complejidad temporal este cerca del uno haciendo la inserción la más eficiente posible

- Conforme la tabla se va llenando (α se aproxima a 1), el núme de muestreos lineales aumenta dramáticamente

α	$\frac{1}{2}(1+\frac{1}{1-\alpha})$	$\frac{1}{2}(1+\frac{1}{(1-\alpha)^2})$
50%	1.5	2.5
75%	2.5	8.5
90%	5.5	50.5

Como se puede observar la complejidad temporal de nuestra tabla hash O (1+ α) se define mediante α, que mientras menor sea el número de colisiones exista dentro de nuestra tabla mayor será le eficiencia que se tenga ya que el valor de α será cada vez más pequeño y no existirá un gran impacto en el tiempo en el que se realiza.

Y esta eficiencia también se puede observar a la hora de buscar los datos, ya que para poder buscar alguna información de la tabla solamente se tiene que proveer con la llave del dato para poder realizar la operación y obtener el índice del dato haciendo que no se tenga que recorrer toda la estructura y nos dé una complejidad temporal de O(1+ α), dándonos con una de las búsquedas más eficientes que hemos utilizados a lo largo de todas las actividades integradoras superando a las búsquedas que tienen una complejidad temporal logarítmica.

Tabla: Complejidad de los algoritmos de búsqueda.

Algoritmo	Mejor	Promedio	Peor
Búsqueda secuencial	$O(1)$	$O(n)$	$O(n)$
Búsqueda secuencial en arreglos ordenados	$O(1)$	$O(n)$	$O(n/2)$
Búsqueda binaria	$O(1)$	$O(\log_2 n)$	$O(\log n)$

Lo que puedo concluir de esta actividad es que las tablas hash nos ofrecen una de las búsquedas e inserciones más eficientes dentro de las estructuras de datos que hemos utilizados y esto es muy útil a la hora de manejar grandes cantidades de información como las que usamos en esta práctica, ya que se nos facilita el encontrar la información necesaria como lo fue el resumen de toda la información de la ip y también como es de sencillo poder hacer más eficiente la inserción cambiando el tamaño máximo de la tabla hash.

Referencias

- TABLAS HASH. (s. f.). https://ccia.ugr.es/~jfv/ed1/tedi/cdrom/docs/tablash.html
- 2.2 Hashing - Programación, refactoriza tu mente. (s. f.). https://docs.jjpeleato.com/algoritmia/hashing
- GeeksforGeeks. (2023b). Hashing Data Structure. GeeksforGeeks. https://www.geeksforgeeks.org/hashing-data-structure/