Nombre: Josue Salvador Cano Martinez ***Matrícula:*** A00829022

Programación de estructuras de datos y algoritmos fundamentales

Módulo: Estructura de Datos de Conjuntos (códigos hash)

Actividad 5.2: Actividad integral sobre el uso de códigos hash (Evidencia de competencia)

Investigación y reflexión individual

Fecha de entrega: 29 de noviembre de 2020

Objetivo

Realizar una investigación y reflexión en forma individual de la importancia y eficiencia del uso de las tablas hash en una situación problema de esta naturaleza.

Parte 1: Investigación

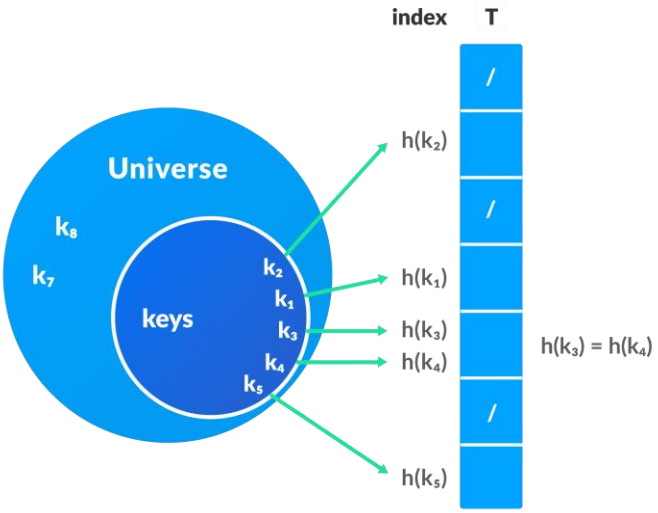
La tabla hash es una estructura de datos que representa datos en forma de pares clave-valor. Cada clave se asigna a un valor en la tabla hash. Las claves se utilizan para indexar los valores / datos. Se aplica un enfoque similar mediante una matriz asociativa.

Los datos se representan en un par clave-valor con la ayuda de claves. Cada dato está asociado con una clave (la clave es un número entero que apunta a los datos).

[1]

Tabla hash

En una tabla hash, las claves se procesan para producir un nuevo índice que se asigna al elemento requerido. Este proceso se llama hash. Sea h(x) una función hash y k sea una clave, h(k) se calcula y se utiliza como índice del elemento.



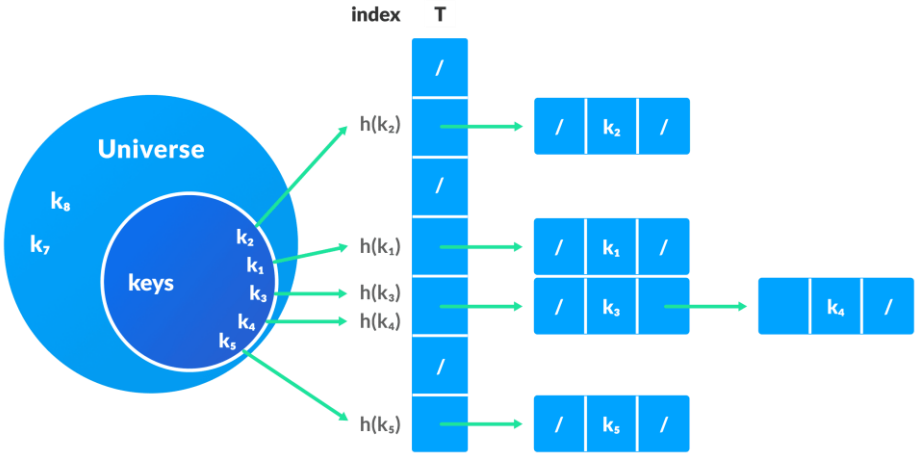
Limitaciones de una tabla hash

Si la función hash produce el mismo índice para varias claves, entonces surge un conflicto. Esta situación se llama colisión.

Para evitar esto, se elige una función hash adecuada, pero es imposible producir todas las claves únicas porque |U|>m. Por lo tanto, es posible que una buena función hash no evite las colisiones por completo, pero puede reducir el número de colisiones. Sin embargo, hay otras técnicas para resolver colisiones.

Resolución de colisiones por encadenamiento

En esta técnica, si una función hash produce el mismo índice para varios elementos, estos elementos se almacenan en el mismo índice usando una lista doblemente enlazada.
Si j es el espacio para varios elementos, contiene un puntero al encabezado de la lista de elementos. Si no hay ningún elemento presente, j contiene NULL.



Método de división

Si k es una clave y m tiene el tamaño de la tabla hash, la función hash h() se calcula como:
h(k) = k mod m

Quadratic Probing

En el sondeo cuadrático, el espaciado entre las ranuras se incrementa (mayor que uno) usando la siguiente relación: donde, h(k, i) = (h′(k) + c1i + c2i2) mod m

Aplicaciones de tabla hash

- Útiles cuando se requiere búsqueda e inserción de tiempo constante
- Útiles para aplicaciones criptográficas
- Útiles cuando se requieren datos de indexación

[3]

Time complexity		
Algoritmo	*Promedio*	*Peor caso*
Espacio	O(n)	O(n)
Búsqueda	O(1)	O(n)
Inserción	O(1)	O(n)
Eliminación	O(1)	O(n)

[2]

Parte 2: reflexión

En esta actividad integradora se llevó a cabo la implementación de una Hash Table para la estructuración de los datos mediante un mapeo, de modo que la complejidad temporal para su acceso se mirase altamente eficiente (complejidad O(1)). En la Hash Table implementada la llave resulta ser la IP, mientras que el valor resulta ser un resumen de dicha IP (el cual incluye el número de accesos, los puertos únicos de dichos accesos y la fecha y hora de estos). El proceso que se siguió para conseguir la implementación programada de dicha solución fue el siguiente:

1) Se fue iterando un fichero con registros, mismos que fueron separados para obtener los parámetros que se buscaban conseguir (ip, puertos, fecha y hora).
2) Utilizando unordered_map se creó una hash table, para la cual, la llave resultó ser un string de la IP y para el valor se utilizó un vector de tipo string, al cual se asignaron los demás parámetros extraídos (puertos, fecha y hora de los accesos).
3) Conforme se iteró el fichero se fue haciendo push_back de los datos correspondientes a los registros en sus índices respectivos definidos por la IP.
4) Al final, sólo se solicita como dato de entrada al usuario un string que resulta ser la IP de la cual se quiere obtener el resumen, a partir de ello es que se accede en tiempo constante a la posición dentro de la Hash Table correspondiente a dicha IP, para posteriormente, con un ciclo for iterar los posibles valores que formen parte del vector albergado en dicha posición.

Con lo anterior se consigue tener una estructura que posibilita guardar más de 16,000 registros y brindar un acceso altamente efectivo al pretender realizar una consulta, cuestión que, para la situación problema, podría suponer ser útil para un análisis rápido basado en consultas de las IP que podrían mirarse vulnerables.

Referencias

[1] Hackerearth, «Hackerearth,» 2020. [En línea]. Available: https://www.hackerearth.com/practice/data-structures/hash-tables/basics-of-hash-tables/tutorial/. [Último acceso: 28 11 2020].

[2] Goodrich, T. Michael, Tamassia y Roberto, «Archive,» 2006. [En línea]. Available: https://en.wikipedia.org/wiki/Hash_table. [Último acceso: 28 11 2020].

[3] Parewa Labs, «Programiz,» 2020. [En línea]. Available: https://www.programiz.com/dsa/hash-table. [Último acceso: 28 11 2020].