Programación de estructuras de datos y algoritmos fundamentales TC1031

REFLEXION 5.2

Nicolas Aguirre v. A00832772

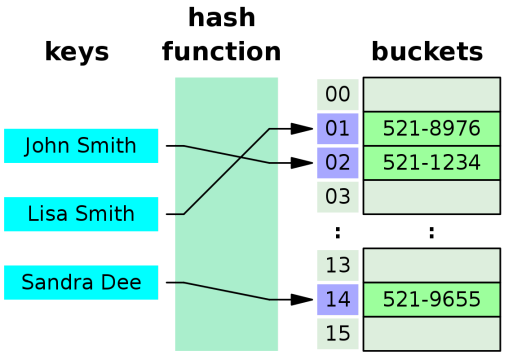
Profesor

Dr. Eduardo Arturo Rodríguez Tello

19 de Junio de 2022

Hashing 5.2 - Reflexión

Las tablas hash son una estructura de datos que usa para almacenar pares de llaves y valores. Estas tablas hash usan una funcion *hash* la cual se usa para calcular un indice en la tabla en donde el elemento va a ser insertado o buscado. El tiempo promedio de complejidad al manipular la tabla hash (como insertar datos, buscarlos o eliminarlos) es de O(1) ya que esta crea la llave en O(1) y accede a esa celda o bucket en O(1) para insertar el dato, de igual manera para borrarlo. En el peor de los casos puede ser O(n). (Garg, n.d.)



Al tener muchos valores, puede suceder que el bucket que la función hash calcula para el elemento, ya este ocupada. Cuando esto sucede, se le llama colision y la funcion debe encontrar otro bucket que no este ocupado. Por ende se debe tener un manejo de colisiones. El manejo de colisiones puede verse con hashing abierto o cerrado. Para el hashing abierto se usa encadenamiento separado en el cual se almacenan los datos con ayuda de una lista la cual se enlaza al indice del bucket calculado, esto se implementa al hacer una funcion llamada *chaining* complejidad O(n) (geeksforgeeks, 2022). En el hashing cerrado, se almacenan los elementos en la misma tabla al buscar el siguiente bucket libre, esto se implementa con la una funcion llamada *quadratic*. La funcion *quadratic* tiene un tiempo de complejidad O(N * L), donde N es la longitud del vector de datos que le demos y L es el tamaño de la tabla hash. (GeeksForGeeks, 2022)

Gracias al hashing, podemos almacenar los datos de las ips de una forma muy eficiente ya que podemos acceder a los datos de las ips de una manera muy rapida ya que a cada ip se le asigna un bucket en especifico. Al tener todas las ips con su información en cada bucket individualmente, podemos acceder a los datos más importantes de las ips como: las aristas saliendo, aristas entrando, y las ips a donde se dirige cierta ip.
El uso del hashing es muy bueno para poder almacenar mucha información y poder accesar muy rapido y eficiente a ella. En este caso, se almacenaron todas las ips y su informacion al transformarlas a un valor entero y luego hacer uso de la función *hashing* con manejo de colisiones *quadratic* para así poder acceder en O(1) a cualquier información de cualquier ip, lo cual es una gran ayuda para tener un programa muy eficiente y rapido.

Bibliografía

Educative. (n.d.). *What is chaining in hash tables?* Educative.io. Retrieved June 19, 2022, from https://www.educative.io/answers/what-is-chaining-in-hash-tables

Garg, P. (n.d.). *Basics of Hash Tables Tutorials & Notes | Data Structures*. HackerEarth. Retrieved June 19, 2022, from https://www.hackerearth.com/practice/data-structures/hash-tables/basics-of-hash-tables/tutorial/

GeeksForGeeks. (2022, February 8). *Quadratic Probing in Hashing*. GeeksforGeeks. Retrieved June 19, 2022, from https://www.geeksforgeeks.org/quadratic-probing-in-hashing/

geeksforgeeks. (2022, May 26). *Hashing | Set 2 (Separate Chaining)*. GeeksforGeeks. Retrieved June 19, 2022, from https://www.geeksforgeeks.org/hashing-set-2-separate-chaining/