Tecnológico de Monterrey

Reflexión 5.2

Alan Ricardo Vilchis Arceo A01640260

13/06/2022

Para esta entrega utilizamos un grafo como la en la entrega anterior, sin embargo ahora cambiamos en tipo de estructura de datos, antes almacenamos los datos en un mapa de la librería estándar de c++, sin embargo ahora utilizamos una hash table, lo cual nos brinda diferentes beneficios, primero nos da una mayor flexibilidad ya que la clase de hash table la estamos creando nosotros por lo tanto la podemos modificar y agregar diferentes métodos que un mapa estándar no. Otra de las mayores diferencias es que la complejidad para encontrar un nodo es de O(1) siempre y cuando no se hayan generado demasiadas colisiones al momento de guardar datos, a diferencia de un mapa el cual usa un red-black-tree que requiere una complejidad de O(log N), por lo tanto es más recomendable utilizar una hash table si el tamaño de los nodos, en este caso registros es muy grande, sin embargo cuando en una hashtable existen muchas colisiones hace que la eficiencia de la estructura aumente a O(N) en su peor caso, ya sea cuando se inserte, borre o busque un elemento, esto es por que el algoritmo va buscando un lugar vacío cuando hay colisiones, y si el número de colisiones es un número cercano a N, es cuando la estructura se hace ineficiente.

Las Hashtable ayudan a la construcción de networks las cuales permite que exista un trackeo de las ips de acceden o salen de un nodo, para este caso, igualmente nos brinda una mayor seguridad al estar utilizando llaves.

Referencias:

GeeksforGeeks. (2022, 12 junio). *Hash Table vs STL Map*.

https://www.geeksforgeeks.org/hash-table-vs-stl-map/

Becky, M. H. (2018, 22 octubre). *Distributed Hash Table Algorithm –*

BitcoinWiki. BitcoinWiki. https://en.bitcoinwiki.org/wiki/Distributed_hash_table