Informe Hashing

Grupo: Los Borbotones

Integrantes:

Huarancca, Antonio

Rivas, Nicolás

Taborda, Gastón

Tema: bucket addressing

Una introducción básica al tema de hashing:

Se usa hashing para mejorar los tiempos de búsqueda y llevarlos a un tiempo de búsqueda óptimo. Por ejemplo, al buscar un registro, el tiempo que se tarde en encontrar este registro tenga orden O(1), lo que significa encontrarlo en el primer lugar en el que se busca. Para esto es necesario hashear el registro, lo que significa asignarle un código o key y guardar las keys de todos los registros de manera ordenada, generalmente en una hash table o tabla de hash. La función de la hash table es guardar los registros de manera ordenada y de acuerdo con sus keys correspondientes.

El problema aparece cuando se tienen uno o más registros con la misma key, por lo cual ocuparían el mismo lugar en la hash table. En éste caso se dice que se produjo una colisión. Existen varios métodos para resolver éste conflicto, uno de los cuáles es el que se trata a continuación: bucket addressing.

Básicamente, consiste en dividir los M slots o posiciones de la hash table en B buckets o cubetas, tal que cada bucket tendrá M/B slots. De ésta manera, al tener varios registros con la misma key, existirá más de una posición en la cual podrían guardarse éstos registros.

Entonces, cuando se requiere buscar un registro en la hash table, lo único que se debe hacer es ir al bucket correspondiente a la key y recorrer los slots del bucket.

|  |
| --- |
| 20 |
| 30 |
| 40 |
| 0 |
| 15 |
| 17 |
| 19 |
| 21 |

Home bucket

***B1 (key=0)*** Donde el registro se divide por la cantidad de

buckets, siendo el resto la key asignada.

Ejemplo: 20%2=0, entonces corresponde al

|  |
| --- |
| 23 |
|  |
|  |
|  |

primer bucket.

***B2(key=1)***

Overflow Bucket de tamaño “infinito”.

Como los slots del bucket que le corresponde

al registro (azul) no tienen ningún slot libre, el

registro se almacena en el overflow bucket.

De ésta manera se optimiza el tiempo de búsqueda, ya que, a pesar de no poder llevarlo al caso ideal de O(1), se logra un orden de O(B), siendo B la cantidad de buckets en los que se dividió cada slot.

También se debe contemplar la situación que se da cuando al hashear un registro se obtiene una key y todos los slots del bucket correspondiente a la key están ocupados. Entonces, recurrimos al overflow bucket.

El overflow bucket es un bucket adicional, común a todos los buckets que se ubica al final de la hash table y tiene tamaño “infinito”. Cuando se produce la situación anterior, se procede al overflow bucket y se guarda el registro en el primer slot libre del mismo.

De ésta manera, el tiempo de búsqueda sufre un incremento en su orden que se corresponde con la cantidad de slots del bucket overflow que se deban recorrer, ya sea al guardar un registro o al buscarlo. Por esto, mientras más registros existan en el bucket overflow, más se incrementará el orden del tiempo de búsqueda.