OBSERVACIONES DEL LA PRACTICA

Maicol Yojan Antonio Rincon código: 202027329

Lindsay Vanessa Pinto Morato Código: 202023139

1. ¿Qué estructura de datos se usa para este índice?

Para la función específica que se pregunta en el laboratorio y que corresponde al siguiente fragmento de código:

**catalog['authors'] = mp.newMap(800,**

**maptype='CHAINING',**

**loadfactor=4.0,**

**comparefunction=compareAuthorsByName)**

Se tiene que la estructura que se realiza es un map, que inicialmente espera guardar 800 autores. El tipo es de tabla de hash sobre la que se implementa es “Separate Chaining” y el factor de carga máximo de la tabla es de 4.0. Esto quiere decir que se espera que la tabla tenga un mínimo de 200 posiciones en las cuales en cada bucket habrá una lista en promedio de 4 elementos.

Así mismo, si analizamos la siguiente función, la cual es similar a la anterior y que se muestra a continuación:

**catalog['bookIds'] = mp.newMap(10000,**

**maptype='CHAINING',**

**loadfactor=4.0,**

**comparefunction=compareMapBookIds)**

Se tiene que la estructura que se realiza es un map, que inicialmente espera guardar 1000 Ids de libros ('bookIds'). El tipo es de tabla de hash sobre la que se implementa es “Separate Chaining” también y al igual que la anterior, el factor de carga máximo de la tabla es de 4.0. Esto quiere decir que se espera que la tabla tenga un mínimo de 250 posiciones en las cuales en cada bucket habrá una lista en promedio 4 elementos.

Ahora bien, si se realiza el mismo análisis para la siguiente función:

catalog['tags'] = mp.newMap(34500,

maptype='PROBING',

loadfactor=0.5,

comparefunction=compareTagNames)

Se tiene que la estructura que se realiza es un map, que inicialmente espera guardar 34500 tags. El tipo es de tabla de hash sobre la que se implementa es “Linear Probing”, el factor de carga máximo de la tabla es de 0.5. Esto quiere decir que se espera que la tabla tenga un mínimo de 69000 posiciones, no obstante, al buscar el siguiente número primo se tiene que el tamaño sería de 69001 posiciones.

1. ¿Cuántos elementos se espera almacenar inicialmente?

Como ya se indicó previamente, inicialmente se espera almacenar un total de 800 autores. Esto puede ser analizado en el fragmento **mp.newMap(800).** Si este valor fuera distinto, la cantidad de autores que se espera almacenar cambiaría también. Este valor debe tenerse en cuenta cuando se analizan los datos en el archivo base. Por ejemplo, si se tuviera la misma función, pero el valor fuera 60, esto querría decir que se espera almacenar 60 datos de autores y si por otro lado, el factor de carga fuera el mismo y se utilizara el mismo mecanismo para las colisiones, se podría inferir que el tamaño de la tabla se esperaría de 15 posiciones.

1. ¿Cuál es el factor de carga?

El factor de carga máximo es de 4.0, esto quiere decir que en cada bucket se tendría una lista con 4 elementos máximo. Este factor de carga se relaciona directamente con el método utilizado, el cual es “Separate Chaining” y por ello puede tomar valores altos (teniendo que el valor de la tabla es 4 veces menor). Por el contrario, si el método utilizado fuera “Linear Probing”, el factor de carga no podría exceder el valor de 0.5 pues por el contrario al anterior, este solo guarda una pareja de llave-valor por posición.

1. ¿Con cuántos elementos serán necesarios agregar para hacer re-hash de la tabla?

Cuando el valor del factor de carga sea mayor a 4.0. Para calcular el valor de la tabla realizamos la siguiente operación:

Ahora bien, en el momento en que el valor pasa de 4.0, es decir 4.1 habría que hacer re-hash, (utilizando aproximación de cifras: 4.06), por lo tanto:

Esto quiere decir que con 812 valores sería necesario hacer re-hash.

1. ¿Qué hace la instrucción “**mp.put(...)”**?

La función mp.put(…) ingresa una pareja de llave-valor a la tabla de hash y si la llave ya existe en la tabla, este valor se reemplaza. Para la función “mp.put(catalog['bookIds'], book['goodreads\_book\_id'], book)” se ingresa una pareja con llave = book['goodreads\_book\_id'] y valor = book.

1. ¿Qué papel cumple **“book[‘goodreads\_book\_id’]”** en esa instrucción?

Es la llave con la cual se va a relacionar el valor. Es decir, es la entrada del índice que conduce al valor asignado.

1. ¿Qué papel cumple **“*book”*** en esa instrucción?

Es el valor al cual se accede desde la llave especificada en el punto anterior.

1. ¿Qué hace la instrucción **“mp.get(…)”**?

Es la función encargada de retornar la pareja llave- valor. Dado un valor específico para la llave, retorna el valor correspondiente a la misma.

En el caso del fragmento de código mostrado y que corresponde a: mp.get(catalog['years'], year), la función retorna el valor (el listado de libros) correspondiente al año ingresado.

1. ¿Qué papel cumple **“*year”*** en esa instrucción?

Year es la llave que se utiliza en el índice para obtener el valor deseado.

1. ¿Qué hace la instrucción **“me.getValue(…)”**?

Retorna el valor de una pareja de un Map. Para el caso de la función presentada, correspondiente a: me.getValue(year)['books'] retorna el valor correspondiente al valor especificado por la llave year.