Mapas y Diccionarios

Emmanuel Cruz Hernández
emmanuel_cruzh@ciencias.unam.mx

8 de diciembre de 2020

Contenido

1. Introducción
2. Tablas Hash
3. Función hash
4. Colisiones
5. Bibliografía

Mapas

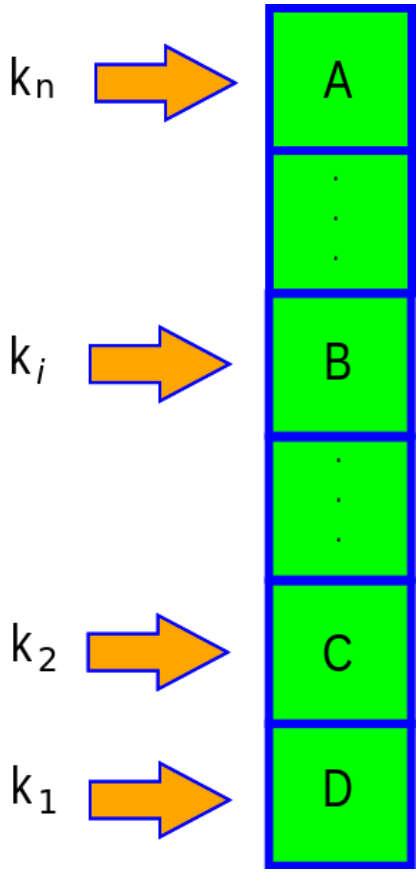
Un mapa es tipo de datos abstracto diseñado para almacenar y recuperar valores de manera eficiente en función de una clave de búsqueda de identificación única para cada uno.

Relación de valores

Específicamente, un mapa almacena pares clave-valor (k, v), que llamamos entradas, donde k es la clave y v es su valor correspondiente.

Se requiere que las claves sean únicas, de modo que la asociación de claves a valores defina una asignación.

Representación gráfica



Operaciones

- size(): regresa la cantidad de elementos contenidos en el mapa.
- isEmpty(): verifica si el mapa tiene elementos.
- get(k): regresa el elemento con clave k. Si no existe tal elemento se regresa null.

Operaciones

- put(k, v): Si el mapa no tiene una entrada con clave igual a k, entonces agrega la entrada v y regresa null; de lo contrario, reemplaza con v el valor existente de la entrada con la clave igual a k y devuelve el valor anterior.
- remove(k): remueve el elemento con clave k y regresa el elemento. En caso de no existir, regresa null.

Tablas Hash

Una tabla hash es una estructura para implementar de forma eficiente un mapa.

Una tabla hash almacena sus elementos usando una llave k para asignarle una entrada a un elemento.

Características de una tabla hash

La asignación de los elementos está dada por una función que calcula la entrada correspondiente a un objeto.

A esta función se le conoce como **función hash**.

Función Hash

El objetivo de una función hash, h, es asignar cada clave k a un número entero en el rango [0, N - 1], donde N es la capacidad del arreglo para una tabla hash.

Enfoque de una función hash

La forma en en que la función hash asigna una entrada específica en un arreglo es recibiendo como parámetro la llave k y calculando la entrada correspondiente según la función.

El valor v se almacena en la entrada $h(k)$.

Una función hash se divide en dos partes: hash code y función de compresión.

Hash Code

Un objeto tiene una llave k que se conoce como *hash code* o *código hash*. Este código está asociado a un objeto en particular.

Este código o valor está contenido en el dominio -∞ a ∞.

Función de compresión

La longitud de una tabla hash está definida por el tamaño del arreglo que la representa.

Dado que el hash code arroja un valor que no está dentro del rango [0, N], la función de compresión tiene el objetivo de comprimir este valor para que entre en el rango correspondiente.

Forma de función hash

La función hash tiene la siguiente forma:

$h(k) = hascode \,\%\, N$

Donde N es el tamaño de la tabla hash.

Colisiones

Una *colisión* es la asignación de dos elementos a una misma entrada en una tabla hash.

Cuando un valor se quiere almacenar en una tabla hash se busca evitar las colisiones. Decimos que una función hash es *"buena"* si mapea las claves en nuestro mapa de tal forma que se minimizan las colisiones.

Tamaño adecuado de una tabla hash

Supongamos que el tamaño de la tabla es un número no primo. Por ejemplo $N = 845$.

Supongamos, además, que tenemos tres valores: A, B, C. Cuyos hash codes son 147, 2682, 1837.

Cálculo de entrada en la tabla

Haciendo las operaciones tomando como base $h(k) = hashcode \,\%\, N$ con N = 845 y hashcodes 147, 2682, 1837 tenemos lo siguiente:

- h(k) = 147 % 845 = 147
- h(k) = 2682 % 845 = 147
- h(k) = 1837 % 845 = 147

Como se puede ver en los ejemplos, los objeto A, B y C corresponden a la misma entrada de la tabla. Hay colisiones.

Búsqueda de una N adecuada

Como se mostró en el ejemplo anterior, 845 no fue una buena elección para asignar a N.

Probemos con $N = 853$.

Cálculo con N=853

Teniendo los hashcodes 147, 2682 y 1837:

- $h(k) = 147 \,\%\, 853 = 147$
- $h(k) = 2682 \,\%\, 853 = 123$
- $h(k) = 1837 \,\%\, 853 = 131$

En este ejemplo se puede observar que 853 fue una mejor elección para N, ya que no se produjeron las colisiones que anteriormente se presentaban.

¿Cuál es la diferencia entre el primero valor y el segundo valor que se asignó a N?

Evitar colisiones

Un punto importante que se debe considerar para evitar colisiones es el tamaño del arreglo. Debe tener las siguientes características.

- El tamaño N del arreglo debe ser un número primo.
- El tamaño N del arreglo debe ser un número grade.

Bibliografía

Goodrich, M.T., Tamassia, R. y Goldwasser, M.H., *Data Structures and Algorithms in Java*, Wiley, Sexta Edición, 2014.

Weiss, M. A., *Data Structures and Algorithm Analysis in Java*, Pearson, Tercera Edición, 2012.