FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

TDA HASH

Lucas Franciulli

[7541/9515] Algoritmos y Programación II

Segundo cuatrimestre 2021

Nombre:	Franciulli, Lucas
Padrón:	107059
Email:	lfranciulli@fi.uba.ar

Índice

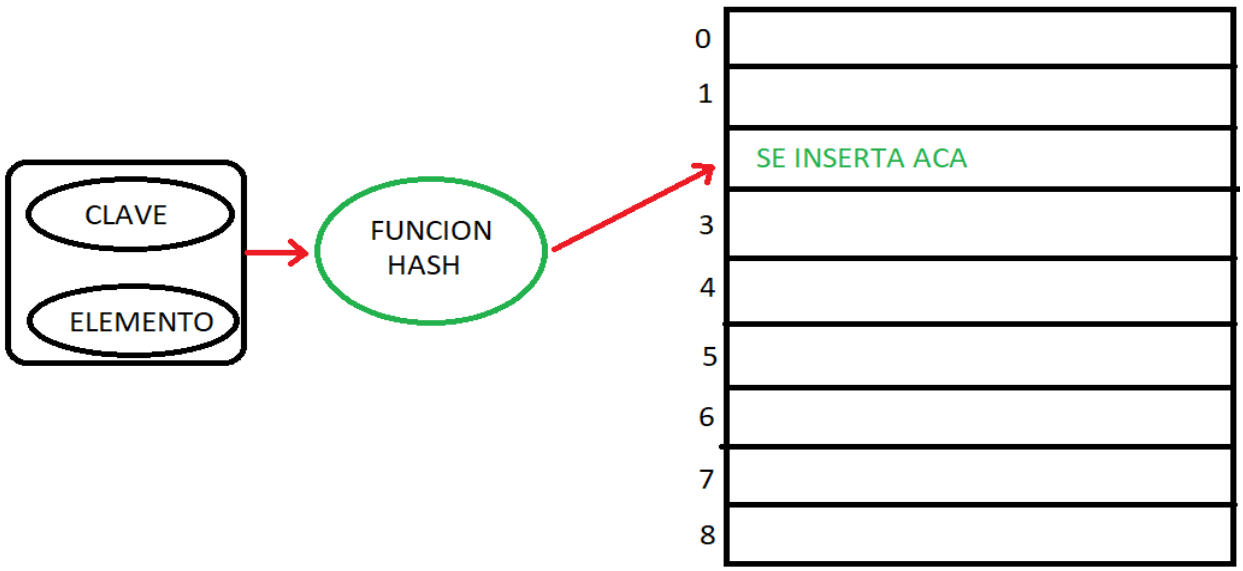
1. Introducción

El objetivo de este informe es responder ciertas preguntas teóricas y explicar el funcionamiento de ciertas funciones del TDA Hash.

2. Teoría

HASH:

Un Hash es un tipo de dato abstracto el cual está conformado por una tabla Hash y una función Hash. Una tabla Hash es una estructura la cual guarda elementos y estos pueden ser encontrados mediante una clave. Una función Hash recibe una clave y devuelve un numero asociado y este debe ser replicable, con esto me refiero a que si yo paso la misma clave varias veces (sin alterar el tamaño de la tabla Hash) el numero que me devuelve la función debe ser siempre el mismo. La idea principal de todo esto es recibir una clave asociada a un elemento, el cual se quiera insertar, pasar esta clave por la función Hash que nos devuelve un lugar en la tabla Hash e insertar la clave y el elemento.



El beneficio principal de Hash es poder mediante una clave aplicar la función Hash y encontrar donde se guardo previamente el elemento que buscamos bajando la complejidad de la búsqueda notablemente.

Además de todo esto existen dos tipos de tabla de Hash: Hash abierto y cerrado.

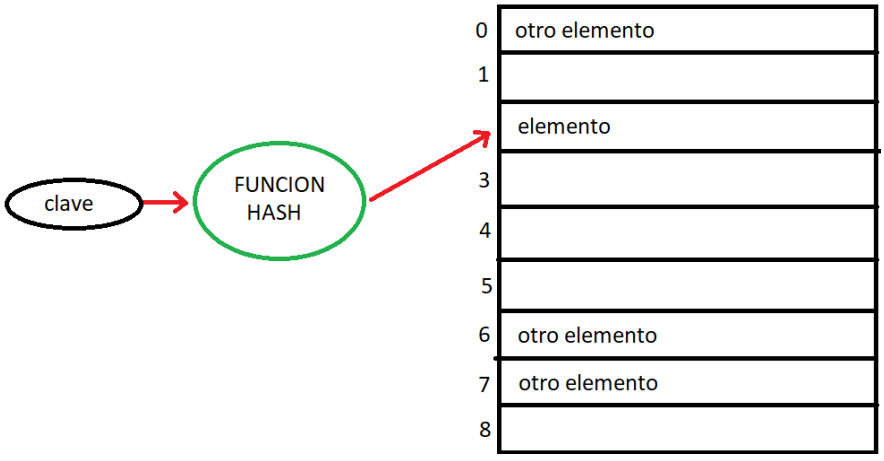
HASH CERRADO (Direccionamiento abierto):

El Hash cerrado se caracteriza por que todos los valores almacenados se guardan en la tabla directamente. Debido a esto el tamaño de la tabla debe ser siempre mayor o igual al número de claves. Otra característica es en el caso de una colisión, ósea cuando dos claves devuelven el mismo numero al aplicarles la función de Hash. Existen varias soluciones a este caso en los Hash cerrados:

-Probing Lineal: Simplemente recorre la tabla Hash hasta encontrar un espacio libre.

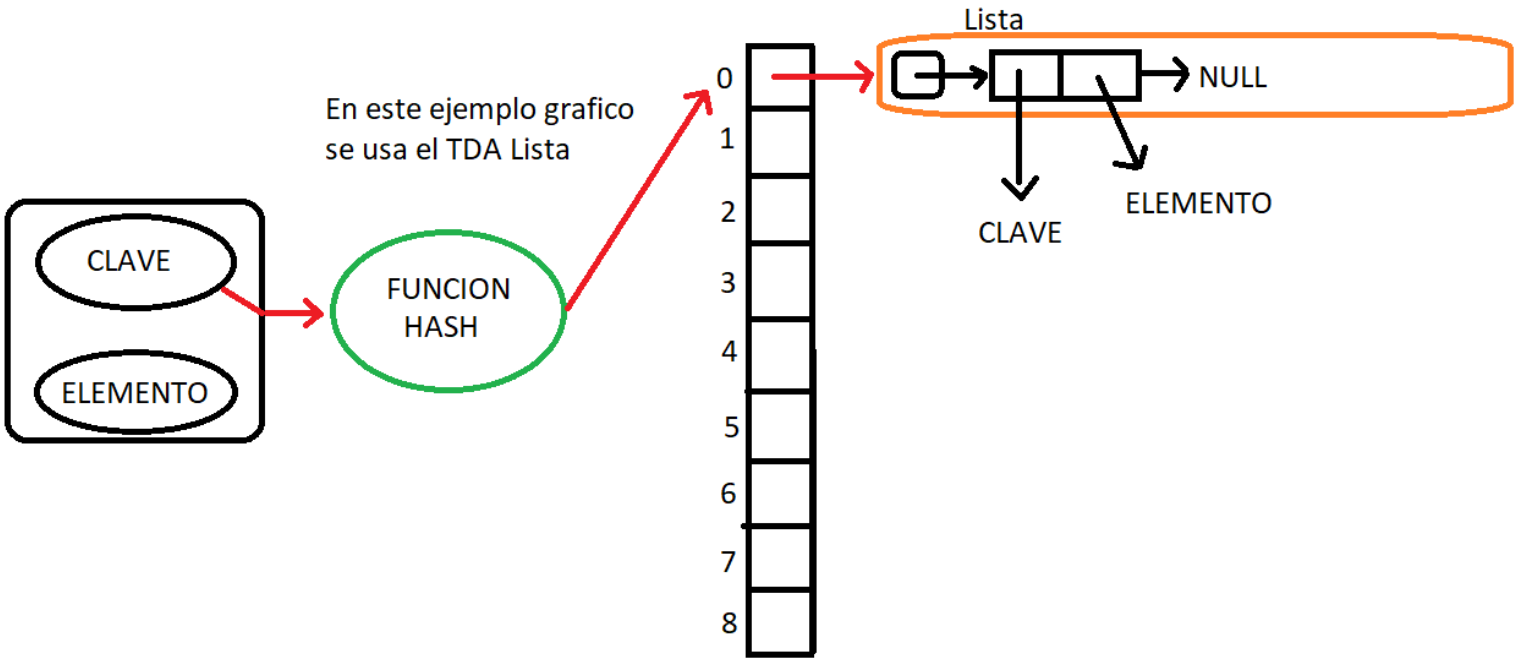
-Probing Cuadrático: (intentos fallidos)^2 para intentar insertar.

-Hash Doble : Aplica una segunda función de Hash en el caso de colisiones.



HASH ABIERTO (Direccionamiento cerrado):

El Hash abierto se diferencia principalmente porque cada celda de la tabla de Hash esta apuntando a un TDA. Este cambio nos lleva a que en el caso de las colisiones no se tenga que recorrer la tabla ni nada por el estilo, en cambio lo que se hace en simplemente añadirlo al TDA designado.



Tanto como en el Hash abierto como en el Hash cerrado existe algo llamado factor de carga que sirve para saber cuando la tabla Hash esta ocupada a cierto porcentaje establecido y que se debe rehashear. Esto significa aumentar el tamaño de la tabla Hash y reinsertar todos los elementos previamente insertados.

3. Funciones

¿Por qué use el TDA Lista en mi Hash Abierto?

La razón principal por lo cual decidí usar el TDA Lista en mi Hash es porque en el momento de una colisión yo debía ver si hay un elemento con una clave igual a la que yo debo insertar por ende debía comparar todas las claves. Una vez teniendo esto presente me pareció que el TDA Lista era el más simple para trabajar con esto.

ESTRUCTURAS UTILIZADAS:

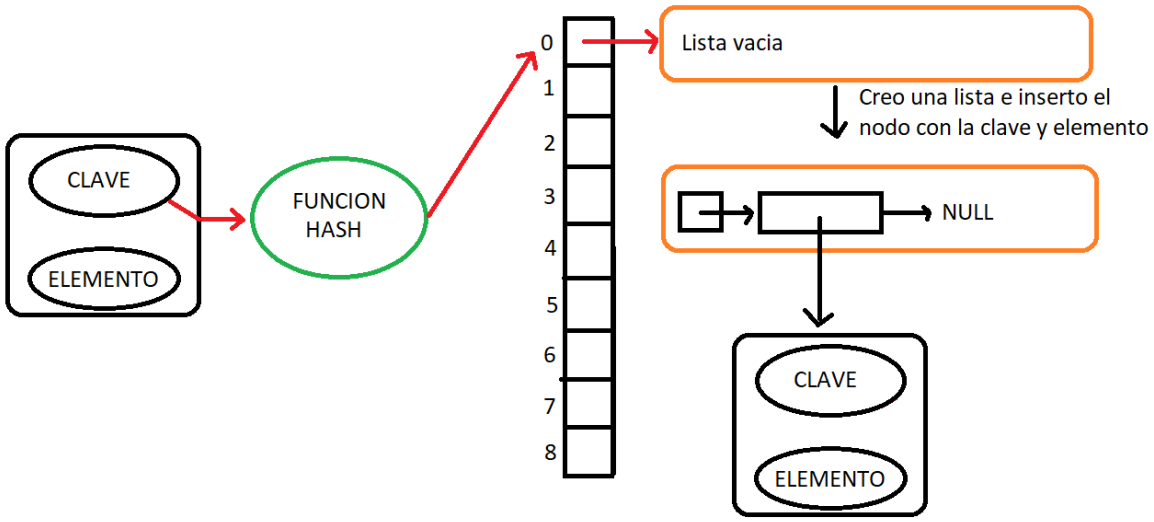
Las estructuras que se usan en el hash.c son básicamente dos. Por un lado, esta la principal de Hash almacena la cantidad de listas utilizadas, la cantidad de casilleros disponibles en la tabla Hash, la cantidad de elementos insertados, el destructor y una lista_t** lo cual nos permite tener un vector dinámico de lista_t*. Por el otro lado se encuentra la estructura del nodo la cual es la que utilizamos para insertar la clave y el elemento que queremos en la lista.

hash_insertar():

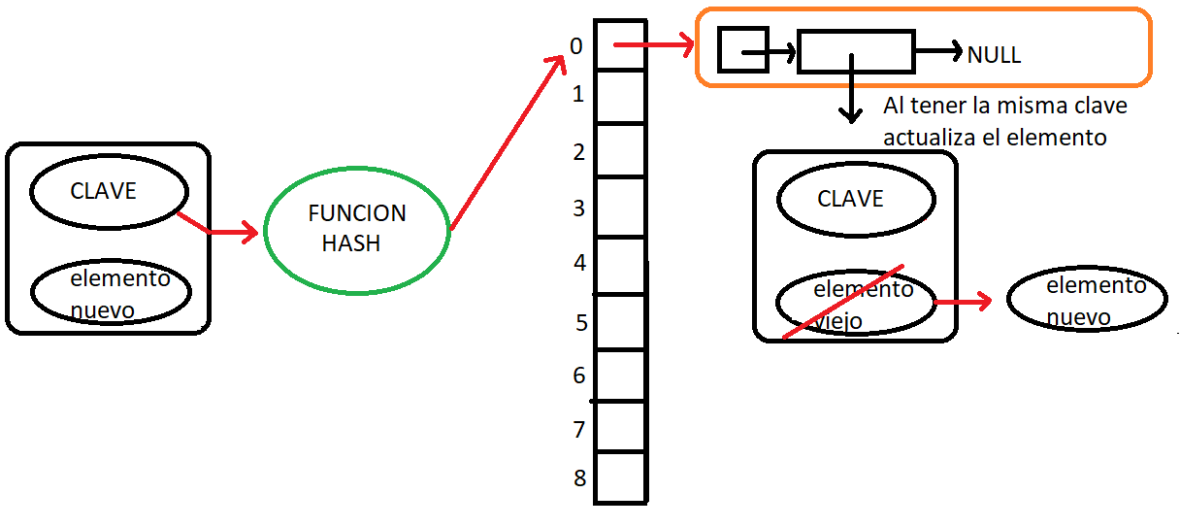
La primera función que vamos a explicar va a ser hash_insertar(). Esta se encarga de recibir una clave, un elemento y un hash e insertar las dos primeras en este último. Para poder lograr esto lo que hace es verificar que los parámetros enviados sean los correctos. Luego copia la clave. Verifica si debe rehashear, ósea que si supero el umbral de rehasheo que es 75%. De ser así llama a la función rehasheo_lista() la cual será explicada a más detalle

más adelante. Una vez terminada esta función calcula el índex del hash en el cual debe ir el elemento que queremos insertar mediante aplicarle la función clave_hash() a la clave dada. Por ultimo se llama a la función insertar_en_lista_hash() la cual se encarga de crear una lista e insertar la clave y elemento si es que no existe una lista, actualizar el elemento si la calve ya existe en la lista o insertar al final de esta si la clave es diferente al resto.

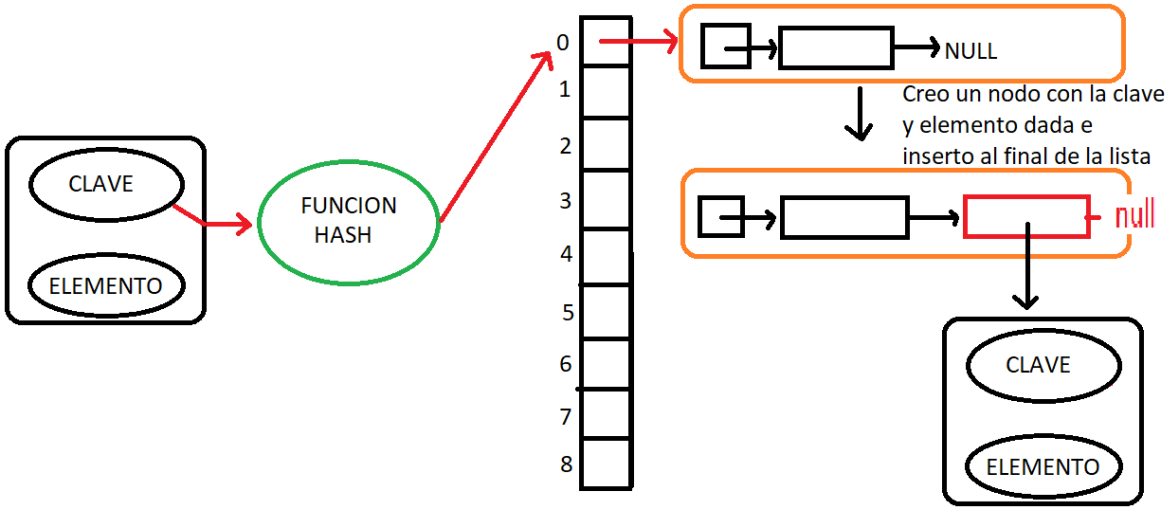
Insertar lista vacía:



Insertar clave repetida:



Insertar al final:



rehasheo_lista():

La función rehasheo_lista() se encarga de crear una tabla Hash nueva, reemplazarla por la vieja sin perder el puntero a esta, recorrer la tabla anterior y reinsertar los elementos de la tabla vieja en la nueva aplicando obviamente la función de hash en el medio la cual es calve_hash().

hash_quitar():

La función de hash_quitar() se encarga de encontrar el elemento que este asociado a la clave que se pasó por parámetro y eliminar ambos de la lista. Para lograr esto encuentra el índex de la clave en el Hash, de ahí recorre la lista preguntándose si en el nodo actual hay una clave igual a la buscada y al final devuelve la posición de esta si fue encontrada o devuelve la posición del ultimo elemento. Como no sabemos si la posición es la del elemento que buscamos nos preguntamos después de obtener el nodo de la posición indicada si la clave es igual a la buscada, de no ser así termina la función ahí, pero de ser la buscada quitamos el nodo de la lista y le aplicamos el destructor del Hash.