Universidad EAFIT
Curso Estructuras de Datos y Algoritmos
Proyecto Final
Luis Eduardo Giraldo Jiménez
Karen Melissa Gomez Montoya

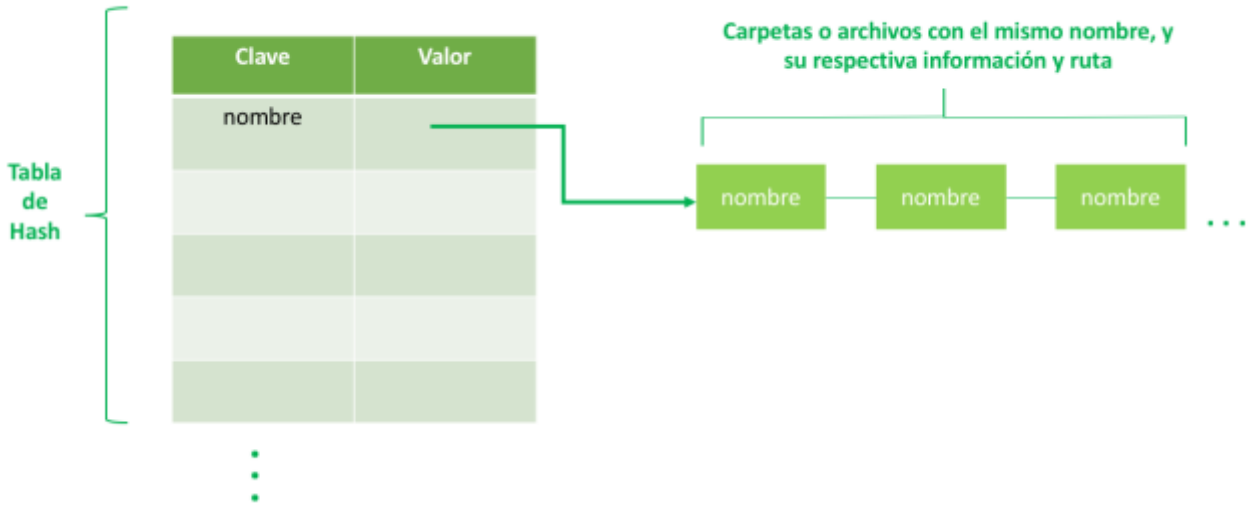
ESTRUCTURA DE DATOS PARA LISTAR Y BUSCAR EL CONTENIDO DE UN DIRECTORIO

El ordenamiento de archivos para su posterior búsqueda representa un problema importante en el área de desarrollo de software. Es relevante escoger una estructura de datos que tenga suficiente capacidad y que permita encontrar con rapidez y eficiencia cada archivo.

El objetivo de esta práctica es desarrollar una estructura de datos para representar los directorios y archivos en un sistema de archivos, y permitir consultar eficientemente los archivos y subdirectorios que se encuentren en un directorio.

Solución

La estructura de datos consiste en una tabla de hash con listas enlazadas. Una tabla de hash “es una estructura de datos que asocia llaves o claves con valores”. Entonces en este caso cada “nodo” de la tabla de hash contiene una lista enlazada con la información de las carpetas con el mismo nombre, siendo este la clave asociada.



¿Por qué hemos elegido esta estructura? Si comparamos con otras estructuras vemos que ésta presenta varias ventajas. Usar listas enlazadas haría que los

tiempos de búsqueda sean mayores, además es más engorroso encontrar los archivos. Las pilas y colas son casos específicos de la lista enlazada por lo tanto no son la herramienta más apropiada. Los árboles, aunque una buena opción, presentan algunas complicaciones en la ejecución: Si lo pensamos como un nodo padre (la carpeta que tiene todas las carpetas) y sus hijos, cada carpeta que hay en su interior, y sus nietos, las carpetas que hay en el interior de los hijos, así hasta llegar a los archivos (que serían las hojas), ya tenemos un problema, al tener tantas carpetas se formaría un árbol gigante y difícil de recorrer, sería difícil también adjudicarle una clave o modo de clasificación.
La tabla de hash tiene una característica especial que la hace bastante útil en nuestro caso, tiene una clave que permite identificar sus elementos.

Respecto a la Complejidad

Las características de la tabla de hash la hacen muy eficiente en cuestión de complejidad, específicamente nuestra estructura de datos disminuye en este aspecto en la situación de que haya muchas carpetas con el mismo nombre lo que en la práctica no tiene mucho sentido. Al buscar por nombre, la tabla de hash va rápidamente a la “caja” que contiene la lista de archivos o carpetas con este nombre y como no suelen ser demasiados en poco tiempo los imprime.

Algunos de los métodos principales del programa son:

modificarDireccion, el cual es elemental en la gestión del archivo, para estructurar las rutas. Este método es de O(n), siendo n el número de carpetas o archivos con el mismo nombre.

listarContenido, este método lista todos los elementos de una carpeta, sean archivos u otra carpeta. Es de O(n).

get (obtener), este método se encarga de buscar y retornar un elemento, ya sea archivo o carpeta, por medio de la clave (nombre de la carpeta) y la ruta del archivo que es lo que nos permite, en parte, diferenciar una carpeta de otra. Este método es de O(n).

put (añadir), este método nos permite ingresar un dato a la tabla de hash. Es de O(1).

La siguiente tabla muestra algunos tiempos de ejecución.

Operación	Mejor tiempo	Peor tiempo	Tiempo promedio
Leer Archivo	653 ms	1.9 s	1.3 s
buscar con nombre	0 ms	0.04 ms	0.00 ms
buscar con ruta	0.07ms	0.23 ms	0.013ms
imprimir	0.87ms	4ms	2.8 ms