

## MÉTODOS DE TRATAMIENTO DE COLISIONES

La elección de un método adecuado para resolver colisiones es tan importante como la elección de una buena función hash. Cuando la función hash obtiene una misma dirección para dos claves diferentes, se está ante una colisión.

Algunos métodos más utilizados para resolver colisiones son los siguientes:

- Reasignación
- Arreglos anidados
- Áreas de desborde

### Reasignación

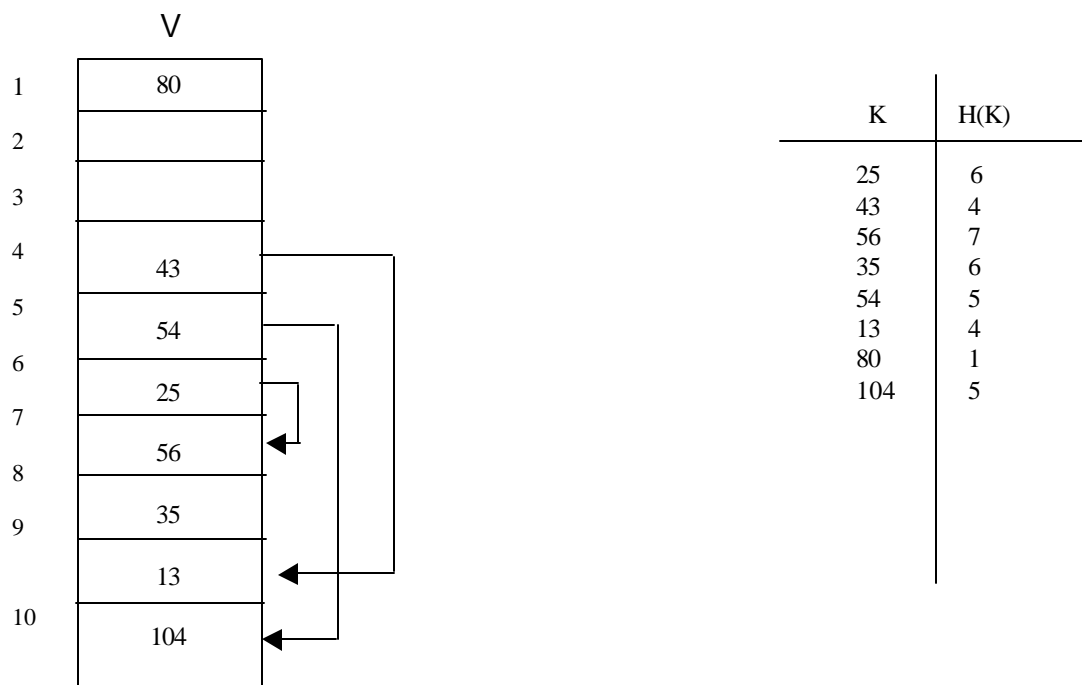
Existen varios métodos que trabajan bajo el principio de comparación y reasignación de elementos. Se analizarán tres de ellos:

- Prueba lineal
- Prueba cuadrática
- Doble dirección hash

#### a) Prueba lineal

Consiste en que una vez detectada la colisión se debe de recorrer el arreglo secuencialmente a partir del punto de colisión, buscando al elemento. El proceso de búsqueda concluye cuando el elemento es hallado, o bien cuando se encuentra una posición vacía. Se trata al arreglo como a una estructura circular: el siguiente elemento después del último es el primero.

La principal desventaja de este método es que puede haber un fuerte agrupamiento alrededor de ciertas claves, mientras que otras zonas del arreglo permanecerían vacías. Si las concentraciones de claves son muy frecuentes, la búsqueda será principalmente secuencial perdiendo así las ventajas del método hash. Con el ejemplo se ilustra el funcionamiento del algoritmo:

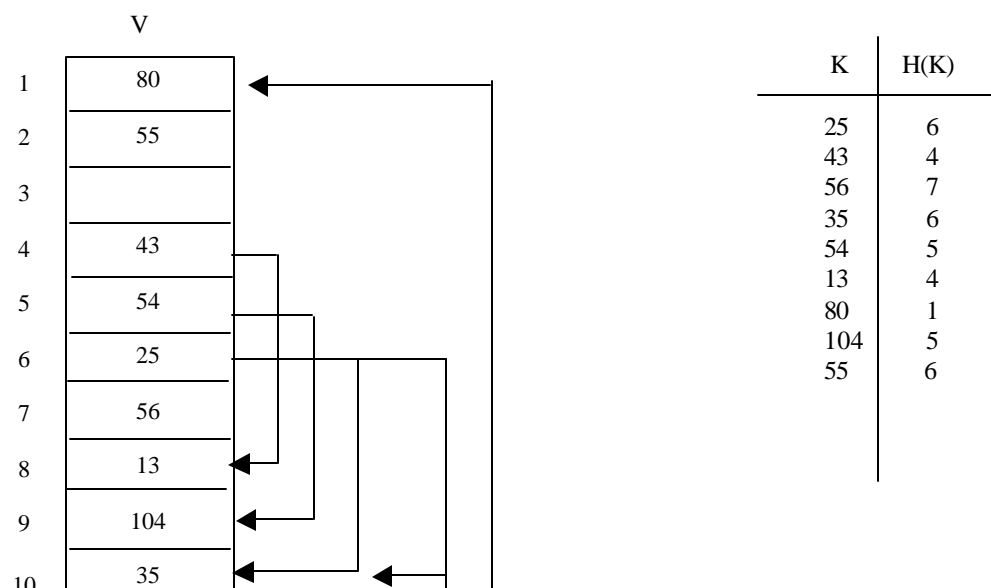


#### b) Prueba Cuadrática

Este método es similar al de la prueba lineal. La diferencia consiste en que en el cuadrático las direcciones alternativas se generan como  $D + 1$ ,  $D + 4$ ,  $D + 9$ ,  $\dots$ ,  $D + i^2$  en vez de  $D + 1$ ,  $D + 2, \dots, D + i$ . Esta variación permite una mejor distribución de las claves colisionadas.

La principal desventaja de este método es que pueden quedar casillas del arreglo sin visitar. Además, como los valores de las direcciones varían en  $i^2$  unidades, resulta difícil determinar una condición general para detener el ciclo. Este problema podría solucionarse empleando una variable auxiliar, cuyos valores dirijan el recorrido del arreglo de tal manera que garantice que serán visitadas todas las casillas.

A continuación se presenta un ejemplo que ilustra el funcionamiento:



### C) Doble dirección hash

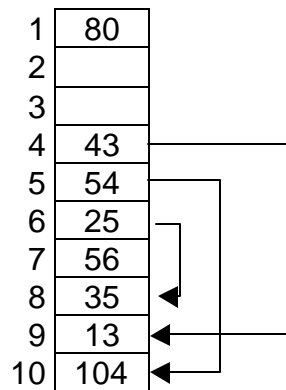
Consiste en que una vez detectada la colisión se debe generar otra dirección aplicando la función hash a la dirección previamente obtenida. El proceso se detiene cuando el elemento es hallado, o bien cuando se encuentra una posición vacía.

$$D = H(K)$$

$$D' = H(D)$$

$$D'' = H(D')$$

La función hash que se aplique a las direcciones puede o no ser la misma que originalmente se aplicó a la clave. No existe una regla que permita decidir cuál será la mejor función a emplear en el cálculo de las sucesivas direcciones.



K	H(K)	H'(D)	H'(D')	H'(D'')
25	6	---	---	---
43	4	---	---	---
56	7	---	---	---
35	6	8	---	---
54	5	---	---	---
13	4	6	8	10
80	1	---	---	---
104	5	7	9	---

## Arreglos anidados

Este método consiste en que cada elemento del arreglo tenga otro arreglo en el cual se almacena los elementos colisionados. Si bien la solución parece ser sencilla, es claro también que resulta ineficiente. Al trabajar con arreglos se depende del espacio que se le asigna a este lo cual conduce a un nuevo problema difícil de solucionar: elegir un tamaño adecuado de arreglo que permita el equilibrio entre el costo de memoria y el número de valores colisionados que pudiera almacenar

V							
1	80			...		k	H(k)
2						25	6
3						43	4
4	43	13				46	7
5	54	104				35	6
6	25	35				54	5
7	56					13	4
8						80	1
9						104	5
10							

## Encadenamiento

Consiste en que cada elemento del arreglo tenga un apuntador a una lista ligada, la cual se va generando e ir almacenando los valores colisionados a medida que se requiera.

K	H(K)
25	6
43	4
56	7
35	6
54	5
13	4
80	1
104	5

