Fundamentos de Organización de Datos

Hashing

Dispersión de Archivos

- ✔ Técnica para generar una dirección base única para una clave dada.
- ✔ Convierte la clave en un número aleatorio, que luego sirve para determinar dónde se almacena la clave.
- ✔ Utiliza una función de dispersión para mapear cada clave con una dirección física de almacenamiento.
- ✔ Utilizada cuando se requiere acceso rápido por clave.

Tipos de Dispersión

Direccionamiento estático

El espacio disponible para dispersar los registros del archivo está fijado previamente.

Direccionamiento dinámico

El espacio disponible para dispersar los registros del archivo aumenta o disminuye en función de las necesidades.

Parámetros a considerar

Parámetros que influyen sobre el desempeño del ambiente de dispersión:

- ✔ Capacidad de almacenamiento de cada dirección
- ✔ Densidad de empaquetamiento
- ✔ Función de hash
- ✔ Método de tratamiento de desbordes

Dispersión de Archivos

Función de dispersión

Caja negra que a partir de una clave genera la dirección física donde debe almacenarse el registro.

Colisión

Situación en la que un registro es asignado, por función de dispersión, a una dirección que ya posee uno o más registros.

Dispersión de Archivos

Desborde

Situación en la cual una clave carece de lugar en la dirección asignada por la función de dispersión.

Densidad de empaquetamiento

Relación entre el espacio disponible para el archivo de datos y la cantidad de registros que integran el mismo.

DE = número de registros / espacio Total

Dispersión de Archivos

Aunque la función de dispersión sea eficiente y la densidad de empaquetamiento sea baja, es probable que ocurran **desbordes**.

Métodos aplicables para resolver colisiones con desborde en *dispersión estática*:

- ✔ ***Saturación progresiva***
- ✔ ***Saturación progresiva encadenada***
- ✔ ***Saturación progresiva con área de desborde por separado***
- ✔ ***Dispersión doble***

No vemos ejercicios prácticos de hashing estático

Hashing Extensible

Técnica de resoluciones: Hashing Extensible

Ejemplo:

- Función de dispersión: Retorna 10 bits.
- Capacidad para 2 registros por dirección.
- Se van a dispersar 8 claves en total.

Hashing Extensible

Claves a dispersar	
1- Colapinto (1011001100)	2- Verstappen (1110101000)
3- Russell (1010001001)	4- Stroll (1010101010)
5- Alonso (1010001000)	6- Hamilton (1001001011)
7- Sainz (1010001111)	8- Leclerc (1010100111)

Hashing Extensible

Estado inicial del archivo:

Tabla de dispersión Bits de dispersión: 0	
Sufijo	#Bloque
(0)	0

Archivo de datos			
#Bloque	Bits	Clave R1	Clave R2
0	0		

Que el número de bits de dispersión esté en 0, indica que no es necesario ningún bit de la secuencia obtenida por la función de dispersión.

Hashing Extensible

Se agregan Colapinto (1011001100) y Verstappen (1110101000)

Tabla de dispersión Bits de dispersión: 0	
Sufijo	#Bloque
(0)	0

Archivo de datos			
#Bloque	Bits	Clave R1	Clave R2
0	0	Colapinto (1011001100)	Verstappen (1110101000)

Ambos se agregan sin inconvenientes en el bloque 0.

Hashing Extensible

Se agrega Russell (1010001001) → produce desborde

Tabla de dispersión Bits de dispersión: 1	
Sufijo	#Bloque
(0)	1
(1)	0

Archivo de datos			
#Bloque	Bits	Clave R1	Clave R2
0	1	Russell (1010001001)	
1	1	Colapinto (1011001100)	Verstappen (1110101000)

Se produce desborde en el bloque 0. Aumentamos en uno los bits de dispersión locales del bloque 0 y creamos un nuevo bloque (bloque 1) con la misma cantidad de bits locales. Luego comparamos los bits locales del bloque con los bits de dispersión globales de la tabla: como ahora el bloque tiene más bits locales que los bits globales de la tabla, aumentamos en uno los bits globales de la tabla y duplicamos la cantidad de direcciones. Finalmente, las claves se redistribuyen entre el bloque original y el nuevo según el bit menos significativo correspondiente. La dirección donde ocurrió el desborde ahora apunta al nuevo bloque.

Hashing Extensible

Se agrega Stroll (1010101010) → produce desborde

Tabla de dispersión Bits de dispersión: 2	
Sufijos	#Bloque
(00)	2
(01)	0
(10)	1
(11)	0

Archivo de datos			
#Bloque	Bits	Clave R1	Clave R2
0	1	Russell (1010001001)	
1	2	Stroll (1010101010)	
2	2	Colapinto (1011001100)	Verstappen (1110101000)

Se produce desborde en el bloque 1. Aumentamos en uno los bits de dispersión locales del bloque 1 y creamos el bloque 2 con la misma cantidad de bits locales. Como los bits locales superan a los bits globales de la tabla, se aumenta en uno los bits de dispersión de la tabla y se duplican la cantidad de direcciones. Finalmente, las claves se redistribuyen entre el bloque original y el nuevo según los dos bits menos significativos. La dirección donde ocurrió el desborde ahora apunta al nuevo bloque.

Hashing Extensible

Se agrega Alonso (1010001000) → produce desborde
Se agrega Hamilton (1001001011) → entra normal
Se agrega Sainz (1010001111) → produce desborde
Se agrega Leclerc (1010100111) → produce desborde

Tabla de dispersión Bits de dispersión: 3	
Sufijo	**#Bloque**
(000)	3
(001)	0
(010)	1
(011)	4
(100)	2
(101)	0
(110)	1
(111)	5

Archivo de datos			
#Bloque	**Bits**	**Clave R1**	**Clave R2**
0	2	Russell (1010001001)	
1	2	Stroll (1010101010)	
2	3	Colapinto (1011001100)	
3	3	Alonso (1010001000)	Verstappen (1110101000)
4	3	Hamilton (1001001011)	
5	3	Leclerc (1010100111)	Sainz (1010001111)

Hashing Extensible

Doy de baja a Verstappen (1110101000): se borra normal

Tabla de dispersión Bits de dispersión: 3	
Sufijo	**#Bloque**
(000)	3
(001)	0
(010)	1
(011)	4
(100)	2
(101)	0
(110)	1
(111)	5

Archivo de datos			
#Bloque	**Bits**	**Clave R1**	**Clave R2**
0	2	Russell (1010001001)	
1	2	Stroll (1010101010)	
2	3	Colapinto (1011001100)	
3	3	Alonso (1010001000)	~~Verstappen (1110101000)~~
4	3	Hamilton (1001001011)	
5	3	Leclerc (1010100111)	Sainz (1010001111)

Al dar de baja a Alonso (1010001000), **el bloque 3 queda vacío**. Este bloque tenía nivel de dispersión local igual al global (3). Se identifica su bloque hermano: aquel que comparte el mismo sufijo salvo en el bit más significativo utilizado para el direccionamiento, en este caso el bloque 2 (sufijo 100). Como el bloque hermano también tiene un nivel de dispersión local de 3 y contiene claves válidas, **se libera el bloque 3** y su rango es absorbido por el bloque hermano. Luego, se reduce en uno el nivel de dispersión local del bloque resultante y se actualizan las entradas de la tabla que apuntaban al bloque eliminado para que redirijan al bloque que permanece.

Tabla de dispersión Bits de dispersión: 3	
Sufijo	**#Bloque**
(000)	2
(001)	0
(010)	1
(011)	4
(100)	2
(101)	0
(110)	1
(111)	5

Archivo de datos			
#Bloque	**Bits**	**Clave R1**	**Clave R2**
0	2	Russell (1010001001)	
1	2	Stroll (1010101010)	
2	2	Colapinto (1011001100)	
3	~~3~~	~~Alonso (1010001000)~~	
4	3	Hamilton (1001001011)	
5	3	Leclerc (1010100111)	Sainz (1010001111)

Al eliminar Stroll (1010101010), el bloque 1 queda vacío. El mismo tenía 2 bits de dispersión local y estaba referenciado por las direcciones de la tabla que terminan en 10. Para saber si se puede liberar el bloque 1, se evalúa si sus direcciones hermanas apuntan al mismo bloque. Son hermanas aquellas que terminan en 00, ya que si se redujera la dispersión local compartirían el mismo sufijo. **Las entradas terminadas en 00 apuntan al mismo bloque (2)**, entonces se puede liberar el bloque 1: se sustituyen las referencias al bloque 1 por referencias al bloque 2, se reduce en uno la dispersión local, y el bloque 1 puede eliminarse. Se mantiene la integridad del direccionamiento y se optimiza el uso de bloques.

Tabla de dispersión Bits de dispersión: 3	
Sufijo	**#Bloque**
(000)	2
(001)	0
(010)	2
(011)	4
(100)	2
(101)	0
(110)	2
(111)	5

Archivo de datos			
#Bloque	**Bits**	**Clave R1**	**Clave R2**
0	2	Russell (1010001001)	
1	~~2~~	~~Stroll (1010101010)~~	
2	1	Colapinto (1011001100)	
3	3	Alonso (1010001000)	
4	3	Hamilton (1001001011)	
5	3	Leclerc (1010100111)	Sainz (1010001111)

Al eliminar Russell (1010001001), el bloque 0 queda vacío. El mismo tenía 2 bits de dispersión local y estaba referenciado por las direcciones de la tabla que terminan en 01. Para saber si se puede liberar el bloque 0, se evalúa si sus direcciones hermanas apuntan todas al mismo bloque. Se consideran hermanas aquellas que terminan en 11, ya que si se redujera la dispersión local compartirían el mismo sufijo. **Las entradas terminadas en 11 no apuntan al mismo bloque** (011 apunta al 4 y 111 apunta al 5), por ende no se puede liberar el bloque 0. **Se escribe el bloque 0 vacío**.

Tabla de dispersión Bits de dispersión: 3	
Sufijo	**#Bloque**
(000)	2
(001)	0
(010)	2
(011)	4
(100)	2
(101)	0
(110)	2
(111)	5

Archivo de datos			
#Bloque	**Bits**	**Clave R1**	**Clave R2**
0	2	~~Russell (1010001001)~~	
1	2	Stroll (1010101010)	
2	1	Colapinto (1011001100)	
3	3	Alonso (1010001000)	
4	3	Hamilton (1001001011)	
5	3	Leclerc (1010100111)	Sainz (1010001111)

Hashing Extensible

Estado final:

Tabla de dispersión Bits de dispersión: 3	
Sufijo	#Bloque
(000)	2
(001)	0
(010)	2
(011)	4
(100)	2
(101)	0
(110)	2
(111)	5

Archivo de datos			
#Bloque	Bits	Clave R1	Clave R2
0	2		
1	2	Stroll (1010101010)	
2	1	Colapinto (1011001100)	
3	3	Alonso (1010001000)	
4	3	Hamilton (1001001011)	
5	3	Leclerc (1010100111)	Sainz (1010001111)