TRANSFORMACIÓN DE CLAVES
HASHING
DISPERSIÓN

ESTRUCTURAS DE DATOS
y ALGORITMOS
LCC – LSI - TUPW

Objetivos

- Conocer la Técnica de Hashing.
- Evaluar el costo computacional de las operaciones, para las distintas políticas de manejo de colisiones.
- Construir el TAD Tabla Hash.

HASHING

Problema

Almacenar y recuperar un elemento particular, identificado por medio de su valor de clave, con el menor *costo* posible.

HASHING

Buscar - Insertar

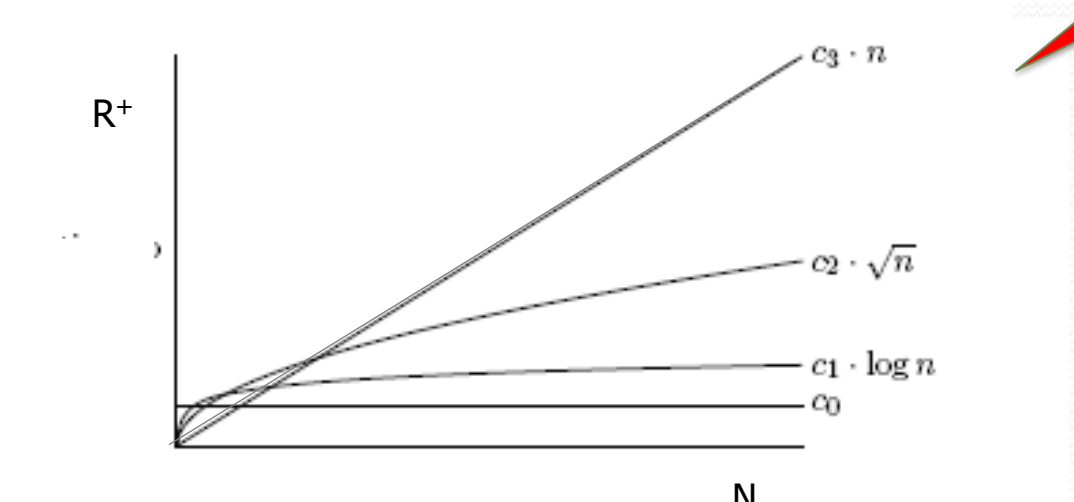
Lista

Árbol B

AVL

ABB

Lista ordenada por contenido



Hashing

HASHING

Los datos se almacenan en direcciones de memoria, entonces:

¿Cómo determinar la dirección de memoria en la cual almacenar cada elemento?

TABLAS DE ACCESO DIRECTO

TABLAS HASH

HASHING

TABLAS HASH

Almacenamiento y Recuperación

$$H : K \rightarrow D$$

transformación –**H**- de claves –**K**- en direcciones –**D**-

HASHING

Si K = { ki / 000 <= ki <= 999}

y |K| = 1000

D = { di / 000 <= di <= 999}

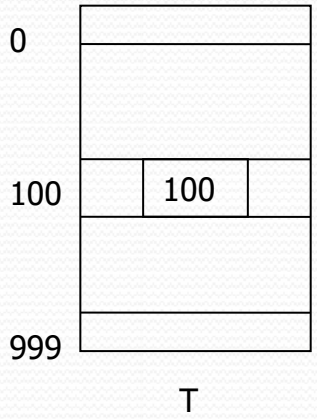
y |D| = 1000

¿Cuál es la función H: K→D ?

Identidad : K → D, esto es:

$h(k_i) = id(k_i) = d$

Si k = 100 entonces id(100) = 100 = d



Funciones biyectivas

En matemáticas, una función

$$F : X \rightarrow Y$$

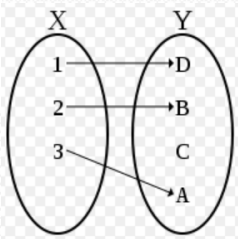
es **biyectiva** si es al mismo tiempo **inyectiva** y **sobreyectiva**

F es **inyectiva** si a elementos distintos del conjunto X (dominio) les corresponden elementos distintos en el conjunto Y (codominio)

$$\forall a, b \in X, \ f(a) = f(b) \Rightarrow a = b$$

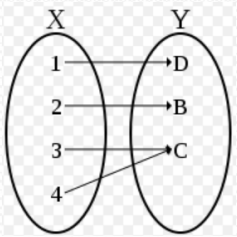
que es equivalente a su contrarrecíproco

$$\forall a, b \in X, \ a \neq b \Rightarrow f(a) \neq f(b)$$ [1]



F es **sobreyectiva** si está aplicada sobre todo el codominio, es decir, cuando cada elemento de Y es la imagen de como mínimo un elemento de X

$$\forall y \in Y \quad \exists x \in X : \quad f(x) = y$$



HASHING

Funciones biyectivas

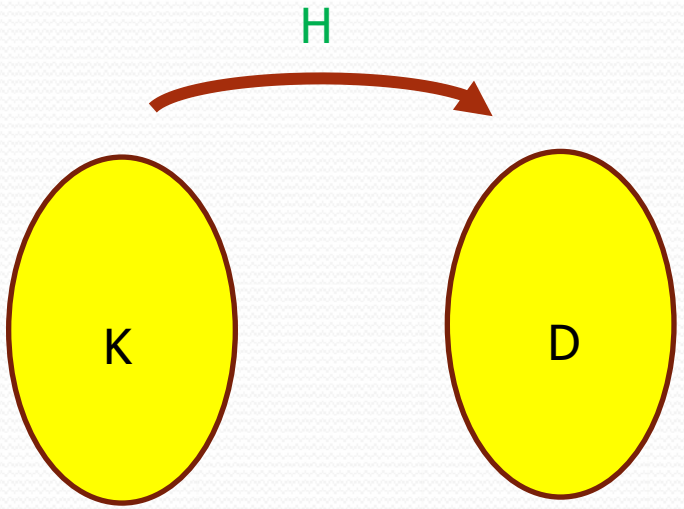
$F : X \rightarrow Y$

- Inyectiva
- Sobreyectiva

$H : K \rightarrow D$

¿Inyectiva?

¿Sobreyectiva?



HASHING

HASHING PERFECTO

- todas las posiciones de la tabla T están ocupadas
- para dos claves distintas k1 y k2, se cumple que h(k1) es distinta de h(k2).

$$k1 \neq k2 \quad \wedge \quad h(k1) \neq h(k2)$$

HASHING

El rango de valores posibles que puede tener **K** (valores distintos de claves) es generalmente mayor que el tamaño de **D**, directamente relacionado con el número de salidas que puede producir **H**.

Problema!!!!

k1 ≠ k2 y **h (k1) = h (k2) = d**

k1 y k2 son claves **sinónimas** que **colisionan** en la dirección **d** bajo la transformación **h**.

La idea…

- Tener una estructura de datos con una complejidad temporal de O(1) en el acceso a los datos
  - Insertar
  - Buscar
  - Borrar
- Las demás operaciones pueden ser O(n)
  - Recorrer
  - Comparar
  - …

HASHING

Aspectos Relevantes

a) Elección de la función de transformación H.

- distribuir las claves uniformemente (todas las direcciones de la tabla tienen la misma probabilidad de ser elegidas)
- ser calculable de modo eficiente (que consigue un propósito empleando los medios idóneos)

Método de la División

Extracción

Plegado

Cuadrado Medio

Funciones aplicables a claves alfanuméricas

b) Política de manejo de colisiones.

Encadenamiento

Uso de Buckets o cubos

Direccionamiento abierto.

HASHING
Funciones de transformación

Método de la División

$h(k) = k \bmod M$

Extracción

Extraer de la clave, los dígitos que varían mas aleatoriamente

Plegado

Si $k_i = k_{i1}\ k_{i2} \mid \ldots \mid k_{in-1}\ k_{in}$

$$h(k_i) = k_{i1}\ k_{i2} + \ldots + k_{in-1}\ k_{in}$$

HASHING

Funciones de transformación

Cuadrado Medio

$k_i = k_{i1}\ k_{i2}\ \ldots\ k_{in-1}\ k_{in}$

$h(k_i) = k_i^{\,2}$ y luego se extraen los dígitos centrales

$d_i = k_{ij}\ k_{ij+1}\ \ldots\ k_{ij+l}$ $\quad 1 < j < j+l < n$ y $\quad 0 <= kij\ kij+1\ \ldots\ kij+l <= M-1$

Funciones aplicables a claves alfanuméricas

$ki = c_{i1}\ c_{i2}\ c_{i3}\ \ldots\ c_{in}$

$h(k_i) = ASCII(c_{i1}) + ASCII(c_{i2}) + \ldots\ldots + ASCII(c_{in})$

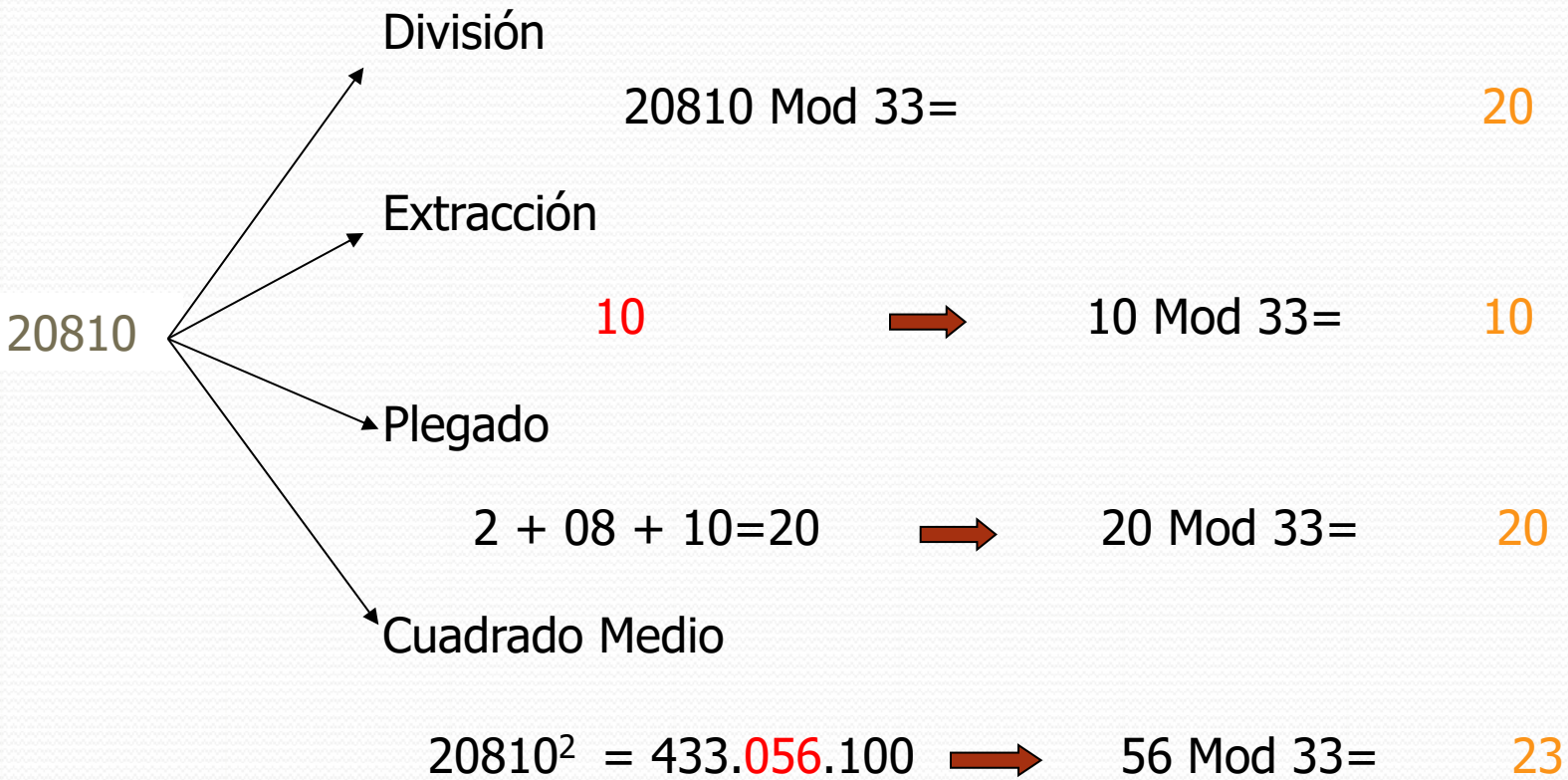
$h(k_i) = ASCII(c_{i1}) * b^1 + ASCII(c_{i2}) * b^2 + \ldots\ldots + ASCII(c_{in}) * b^n$

HASHING

Funciones de transformación

Claves = {20810, 21438, 21478, 21755, 21705, 21762, 21444,…}

$|\text{Claves}| = 33$



HASHING

Funciones de transformación

CLAVES	MOD 33	EXTRACCIÓN	PLEGADO	CUADRADO MEDIO
20810	20	10	20	23
21438	21	5	21	26
21478	28	12	28	7
21755	8	22	8	16
21705	24	5	24	8
21762	15	29	15	23
21444	27	11	27	20
…	…	…	…	…

HASHING

Funciones de transformación

$$h1(k_i) = ASCII(c_{i1}) + ASCII(c_{i2}) + \ldots\ldots + ASCII(c_{in})$$

k : ROMA

DEC	HEX	OCT	CHAR	DEC	HEX	OCT	CH	DEC	HEX	OCT	CH	DEC	HEX	OCT	CH
0	0	000	NUL	32	20	040		64	40	100	@	96	60	140	`
1	1	001	SOH	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT	36	24	044	$	68	44	104	D	100	64	144	d
5	5	005	ENQ	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB	41	29	051	)	73	49	111	I	105	69	151	i
10	A	012	LF	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	B	013	VT	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	C	014	FF	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	D	015	CR	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	E	016	SO	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	F	017	SI	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE	48	30	060	0	80	50	120	80	112	70	160	p
17	11	021	DC1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM)	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS	60	3C	074	<	92	5C	134	\	124	7C	174	|
29	1D	035	GS	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

HASHING

Funciones de transformación

$$h1(k_i) = ASCII(c_{i1}) + ASCII(c_{i2}) + \ldots\ldots + ASCII(c_{in})$$

k_1 : ROMA k_2 : RAMO

Caracter	ASCII
R	82
O	79
M	77
A	65

h1 (ROMA) = 82 + 79 + 77 + 65 =303 = 82 + 65 + 77 + 79 = h1(RAMO)

$$h2(k_i) = ASCII(c_{i1}) * b^1 + ASCII(c_{i2}) * b^2 + \ldots\ldots + ASCII(c_{in}) * b^n$$

h2 (ROMA) = $82 * 10^1 + 79 * 10^2 + 77 * 10^3 + 65 * 10^4$ = 735720

h2 (RAMO) = $82 * 10^1 + 65 * 10^2 + 77 * 10^3 + 79 * 10^4$ = 163320

HASHING

Aspectos Relevantes

a) Elección de la función de transformación H.

- distribuir las claves uniformemente
- ser calculable de modo eficiente

Método de la División

Extracción

Plegado

Cuadrado Medio

Funciones aplicables a claves alfanuméricas

b) Política de manejo de colisiones.

Encadenamiento

Uso de Buckets o cubos

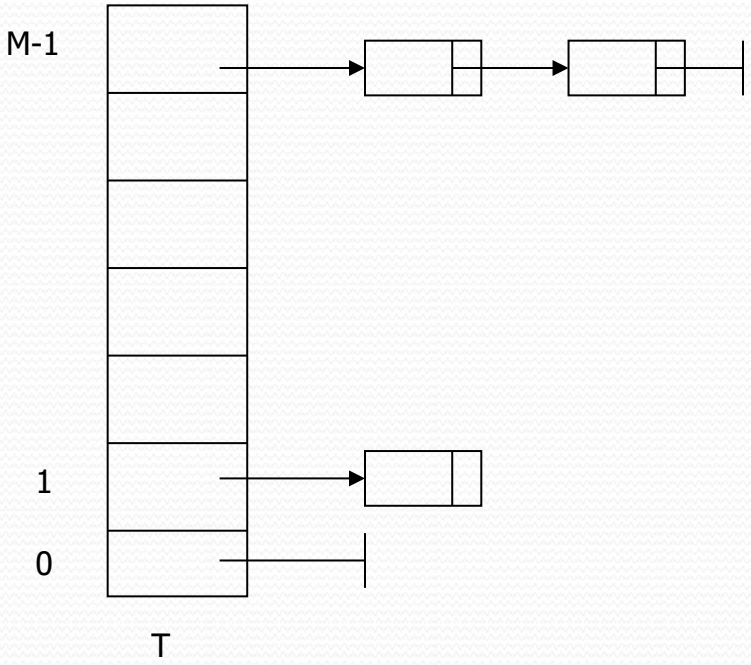
Direccionamiento abierto.

HASHING

Políticas de manejo de colisiones

Encadenamiento o Dispersión Abierta

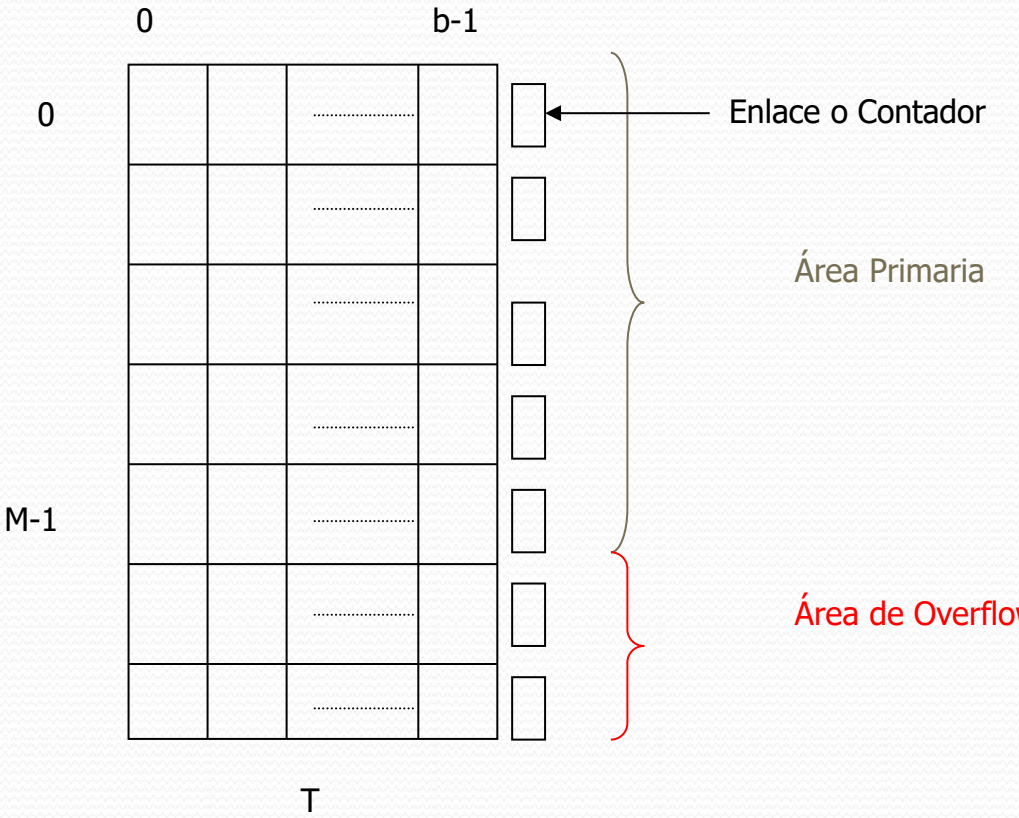
¿ M ?



HASHING

Políticas de manejo de colisiones

Uso de Buckets o Cubos

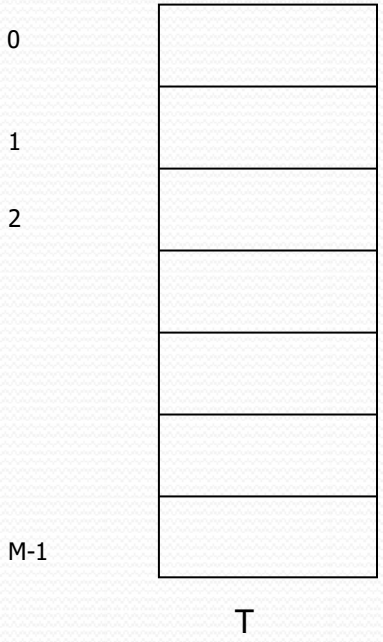


¿ M ?

HASHING

Políticas de manejo de colisiones

Direccionamiento Abierto



Ver Video Direccionamiento Abierto.mp4

HASHING

Políticas de manejo de colisiones

- *Secuencia de prueba lineal*

h(k), h(k)-1,.....,1, 0, M-1, M-2,....., h(k)+1
reducida a módulo tamaño de la tabla

Agrupamientos Primarios!!!

- *Secuencia de prueba pseudo random*

h(k)+0, h(k)+r1, ..., h(k)+rm-1
reducida a módulo tamaño de la tabla

En lugar de r1…rm-1, se puede usar un solo nº aleatorio para toda la secuencia

Agrupamientos Secundarios!!!

- *Doble Hashing*

*h(k) – i * h2 (k) para 0 <= i <= M-1*

HASHING

Factor de Carga

Standish define al ***factor de carga*** α de una tabla como la razón entre el número de entradas N ocupadas en la tabla y el número total de entradas M en la tabla

$$\alpha = N/M$$

Este autor expresa que si α **<= 0.7**, la cantidad de comparaciones involucradas en la recuperación de un elemento es adecuada.

Esta propuesta surge al considerar que la performance se deteriora especialmente al aplicar la secuencia de prueba lineal, a medida que la tabla se aproxima a la saturación total – cuando N se aproxima a M.

HASHING

N=33

M=33

$\Rightarrow$ α = N/M = 1

α = 0.7 y 0.7= N/M

¿ M ?

47 ?

Nros primos

2, 3, 5, 7, 11, 13, 17,
19, 23, 29, 31, 37, 41,
43, **47**, 53, 59, 61, 67,
71, 73, 79, 83, 89 y 97

T

XXXX	0
XXXX	1
XXXX	
	7
21755	8
XXXX	9
	19
20810	20
21438	21
XXXX	22
XXXX	
20551	30
XXXX	
XXXX	M-1

T.A.D. Tabla HASH

¿Objeto de Datos?

¿Operaciones Abstractas?