Topic Hashing in Data Structure Unit No. 04

In data structures,

- Hashing is well known technique to search any particular element among several elements.
- It minimizes the number of comparisons while performing the search.
- Hashing Technique based on Hash Table

Advantage

Unlike other searching techniques, (Linear and binary search)

① Hashing is extremely efficient.

② Time taken by it to performed the search does not depend on the total number of elements.

③ It completes the search with constant time Complexity O(1)

Hashing Mechanism

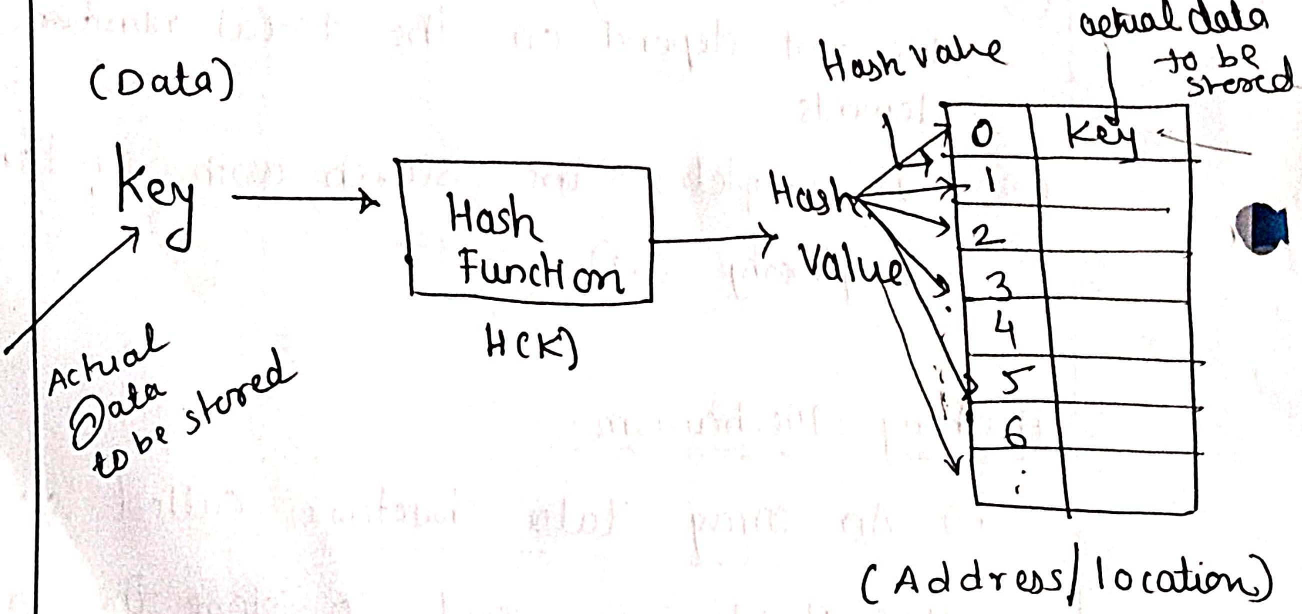
① An array data structure called as Hash Table is used to store the data items.

Topic
Unit No.

④ Based on the hash key value, data items are ensorted into the hash table.

Hash key Value

① Hash key value is a special value that serves as an index for a data item.
② It endicates where the data item should be stored in the hash table.
③ Hash key value is generated using a hash function.



Topic

Unit No.

Hash Function

Hash function is a function that maps any big number or string to a small integer value.

① Hash function takes the data items as an input and returns a small integer value as an output.

② The small integer value is called as hash value.

③ Hash value of the data item is then used as an index for storing it into the hash table.

Types of Hash Functions

There are various types of hash functions available such as,

1. Division Hash Function
2. Mid Square Hash Function
3. Folding Hash Function ~~etc~~.
4. Modulo Multiplication Hash Function

It depends on user which hash Function he want to use.

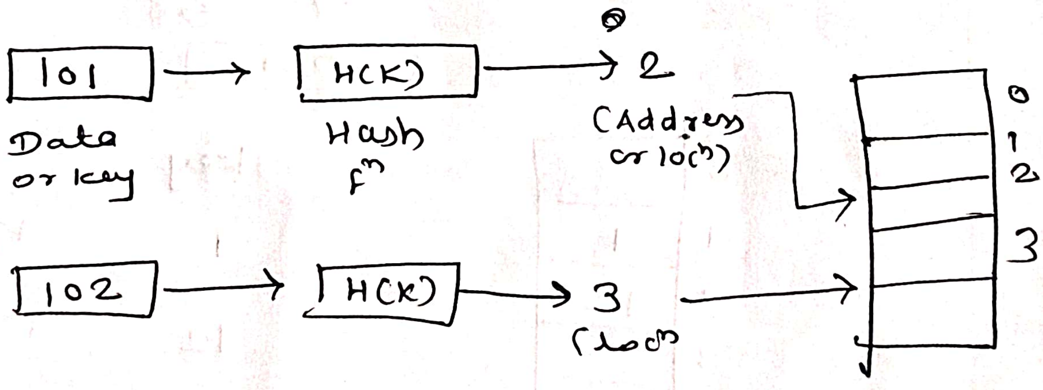
Topic | Unit No.

Types of Hashing Function

Direct Hashing

1. In direct hashing No ~~algorithm~~ Algorithmic calculations are required.
2. No collision occur (Minimum collision)
3. it is limited
4. Not suitable for large key value.

Example



(2) Modulo Division Hashing

- Also known as Division remainder Hashing
- Work with any size of data.

$$\boxed{H(K) = K \bmod n}$$

Where K = key / Data

n = size of list

Example

1, 12, 15, 27, 38.

& $\boxed{n = 10}$

0	
1	1
2	12
3	
4	
5	15
6	
7	27
8	38
9	

key	location
1	1%10 = 1
12	12%10 = 2
27	27%10 = 7
15	15%10 = 5
38	38%10 = 8

Topic　　　　　　　　　　　　　　　　　　Unit No.

- If list size is a prime number then less number of Collision occur.

③ Mid Square Hashing

① Middle square Hashing

② Key is seperated and the address is selected from the middle of square number

③ problem :- Non Uniform key distribution

eg. key = 9452

$= (9452)^2 = 89\underline{340304}$

④ After Mid square we have to take middle number that is depend on Hash table size

Example : ① Size is 1000 then take middle '3' number

② If size is 10000 then take middle '4' number.

Topic Unit No.

Folding Hashing

Fold Shift Hashing	Fold Boundary Hashing
① key value is divided into parts whose size matches to the size of the required address.	In this left & right no are folded on a fixed boundary between them & the center number is calculated)

Fold Shift Hashing

Example

key = 123456789

size = 3

as size = 3

① Divide it into three parts.

$P_1 = 123$

$P_2 = 456$

$P_3 = 789$

② perform Addition of

Result = $P_1 + P_2 + P_3$

= 123
456
789

1368

key value = 1368

As size = 3 Discard the value of 1st number

key value = 368 (Add)

Fold Boundary Hashing

Example 123456 789

step ① . 123 Folded to = 321

& 789 Folded to = 987

middle No (as it is) = 456

② Perform the Add^n.

+ 321
+ 987
+ 456

1764

③ Now Discard the number ①

key value = 764

(IV) Pseudo Random Hashing

$Y = ax + c$ Y = Random No, a = Coefficient

$h(k) = y \bmod n$ (Less chances of collision)

(4) Digit Extraction Method

$M = 1000$

(1) Key = $\underset{P_1}{\underline{789}}\ \underset{P_2}{\underline{456}}\ \underset{P_3}{\underline{123}}$ (as size M = 1000) [take '3' digits for group /Extract]

(2) perform Add^n of P_1, P_2, P_3

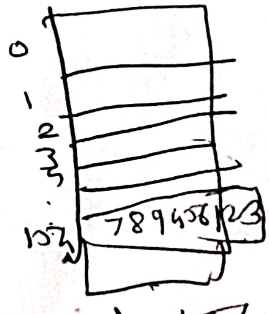
$= P_1 + P_2 + P_3 = 789 + 456 + 123$

key = $\underline{\underline{1368}}$

(3) as. size of Hash table = 1000

& 1368 > 1000 so again Apply

digit Extraction Method



$\therefore$ key = $\underset{P_1}{\underline{136}}\ \underset{P_2}{\underline{\underline{8}}}$

$= P_1 + P_2$

$= 136 + 8$

$= \underline{\underline{154}}$ (Now stored at 154 locn)

Open Hashing

Chaining

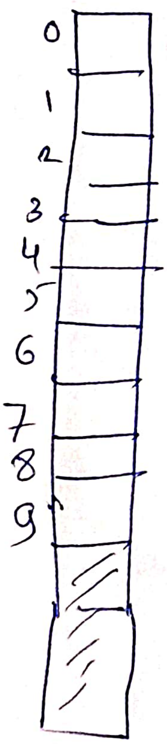
~~Load Fa~~

Keys :

25, 97, 80, 65, 38, 40, 59, 75, 17, 99, 35, 23

Let, $h(k) = k \bmod M$

Load Factor



Each chain (Avg) $\simeq \dfrac{n \text{ (no. of elements)}}{m \text{ (size of hash f}^n\text{)}}$

Load factor

$$\boxed{\alpha = \frac{n}{m}}$$

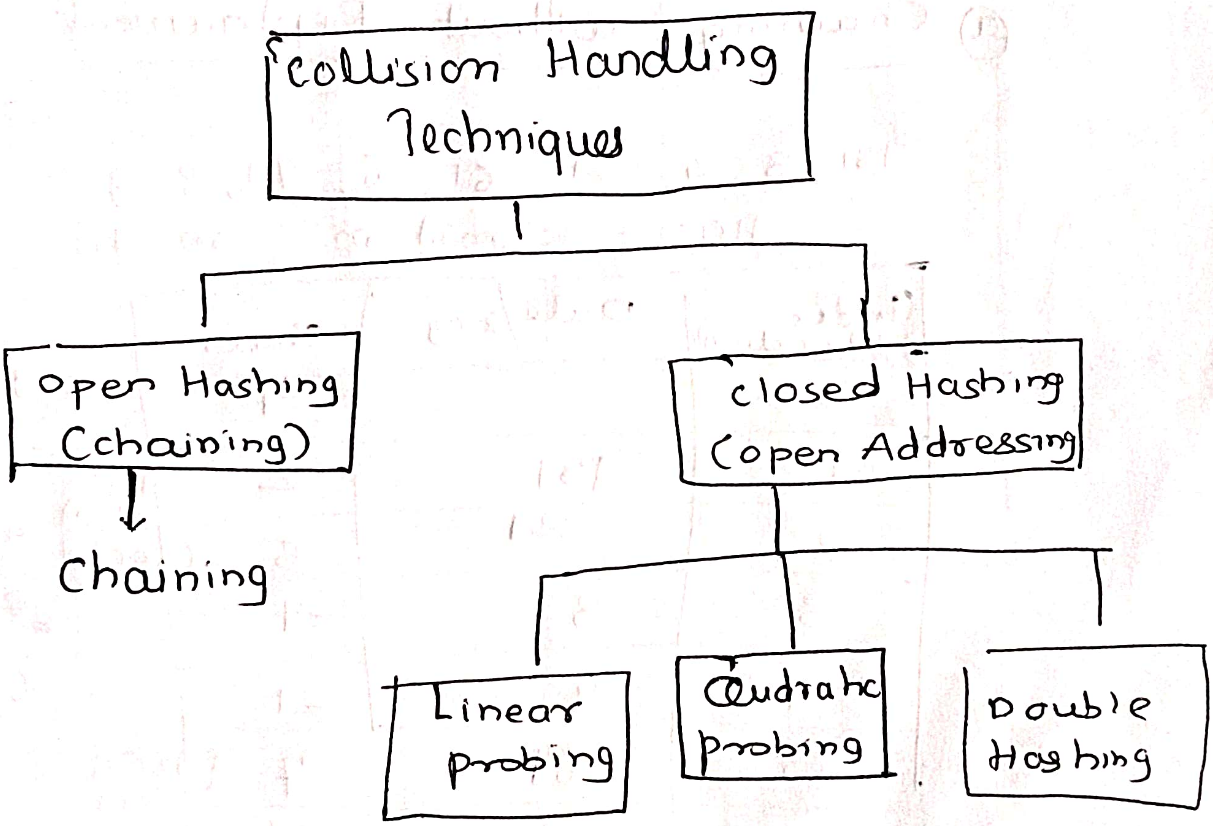
Sanjivani Rural Education Society's
SANJIVANI COLLEGE OF ENGINEERING, KOPARGAON
DEPARTMENT OF COMPUTER ENGINEERING

Topic Unit No.

Simple tP
LP cost ch unh fer
double hashing

Collision Resolution Technique

- Collision occur then it should be handled by applying some techniques, such techniques are called collision handling techniques.



(a) Chaining without Replacement

131, 3, 4, 21, 61, 6, 71, 8, 9

H(K) = K mod m, m = 10

Index (Location)	Data/Key	Chain
0		-1
1	131	2 (loc^n of 21)
2	21	5 (loc^n of 61)
3	3	-1
4	4	-1
5	61	7 [loc^n of 71]
6	6	-1
7	71	-1
8	8	-1
9	9	-1

131 % 10 = 1

3 % 10 = 3

4 % 10 = 4

21 % 10 = 1 (collision)

61 % 10 = 1 (collision)

6 % 10 = 6 ~~(collision)~~

71 % 10 = 1 (collision)

8 % 10 = 8

9 % 10 = 9

Prepared by :

Topic — Unit No. 03

Collision Resolution Technique

Types of Hashing

1. Open Hashing (closed Addressing)
2. closed Hashing (open Addressing)

Types of Hashing

- Open Hashing (closed ~~Hashing~~ Addressing)
  - ↓ Seperate chaining Method
- closed Hashing [open Addressing]
  - ↓ Linear probing
  - Quadratic probing
  - Doubl Hashin

1. open Hashing (closed Addressing)

- In this seperate chaining method is used to ~~avoid~~ solve the collision
- In seperate chaining method seperate chain is maintain along with hash Table whenever collision is occur.

Topic

* Open Hashing (closed Addressing)

Example key = { 3, 2, 9, 6, 11, 13, 7, 12 }

$H(K) = 2K+3$, $m = 10$,

use division method & closed addressing to store these values.

⇒ So, Division method = key % m

as Here, $H(K) = 2K+3$ & $m = 10$

∴ Hash Table

Loc^n	Key
0	
1	9
2	
3	
4	
5	
6	
7	2
8	
9	3

$H(K) = (2K+3)\%m$

Key	~~Location~~	loc^n
3	$= (3\times2+3)\%10$	9
2	$= (2\times2+3)\%10$	7
9	$= (2\times9+3)\%10$	1
6	$(2\times6+3)\%10$	5
11	$(11\times2+3)\%10$	5 C
13	$(13\times2+3)\%10$	9 C
7	$(2\times7+3)\%10$	7 C
12	$(2\times12+3)\%10$	7 C

Prepare the Hash Table.

Loc^n/Address	Key
0	
1	9
2	
3	
4	
5	6 → 11 | /
6	
7	2 → 7 | → 12 | /
8	
9	3 → 13 | /

Advantages:-

1. Simpler to emplement
2. Less Sensitive to Hash function
   - even Hash f^n is not good or poor collision are managed efficiently by using linked list (chaining)
3. Dynamic memory allocation.

Disadvantages

1. Extra space - Required additional space to store linked list
2. poor cache performance
3. worst case mapping- if all element are mapped with same index

* Open addressing (closed Hashing)

* Linear Probing ⇒ Here whenever collision is occur search the next first empty Loc^n is to store the key (value).

Example : Hash Table

key = { 3, 2, 9, 6, 11, 13, 7, 12 }

$H(K) = (2K+3)$

$m = 10$

Use division method & open addressing to store the value.

⇒ Here we can used linear probing method.

∴ $H(K) = (2K+3)$, $m = 10$

∴ Division method = $H(K) \% m$

Key	~~for~~ $H(K) = (2K+3)\%10$	Loc^n	probe
3	$(2\times3+3)\%10 = 9$	9	1
2	$(2\times2+3)\%10 = 7$	7	1
9	$(2\times9+3)\%10 = 1$	1	1
6	$(2\times6+3)\%10 = 5$	5	1
11	$(2\times11+3)\%10 = 5$	5	2
13	$(2\times13+3)\%10 = 9$	9	2
7	$(2\times7+3)\%10 = 7$	7	2
12	$(2\times12+3)\%10 = 7$	7	6

Now, Let us prepare the Hash Table.

Locn	Key
0	13
1	9
2	12
3	
4	
5	6
6	11
7	2
8	7
9	3

∴ Order of Elements in Hash Table.

= { 13, 9, 12, -, -, 6, 11, 2, 7, 3 }

∴ Avg Probe = $\frac{16}{8}$ = 2

$\boxed{\text{Avg. Probe} = 2}$

(2) Quadratic Probing (Open Addressing)

$\boxed{(u+i^2)\%m}$

Here we are moving quadratic way.

Key = {3, 2, 9, 6, 11, 13, 7, 12}

H(K) = (2K+3), m=10

Use division method

⇒

Key	H(k) = (2k+3)% m (=10)	loc^n	probe
3	(2×3+3)%10 =	9	1
2	(2×2+3) % 10 =	7	1
9	(2× 9+3) % 10 =	1	1
6	(2×6+3) % 10 =	5	1
11	(2×11+3) % 10 =	5	2
13	(2×13+3) % 10 =	9	2
7	(2×7+3) % 10 =	7	2
12	(2×12+3) % 10 =	7	5

Avg Probe = 15/8 = 1.8825

as, for Key = 11 collision is occur

$\therefore$ $(u+i^2) \% m$, $i = 0$ to $m-1$.

$\therefore$ $u = 5$

for $i = 0$, $(5+0) \% 10 = 5$

for $i = 1$, $(5+(1)^2) \% 10 = 6$

as for Key = 13 collision is occur

$(u+i^2) \% m$, $i = 0$ to $m-1$

$u = 9$, $m = 10$

$i = 0$, $(9+0^2) \% 10 = 9$

$i = 1$ $(9+(1)^2) \% 10 = 0$

as for key = 7, collision is occur.

$(u+i^2) \% m$ $i = 0$ to $m-1$

$u = 7$, $m = 10$

for $i = 0$, $= (7+(0)^2) \% 10 = 7$

$i = 1$, $= (7+(1)^2) \% 10 = 8$

Now, for key = 12 collision is occur

$\therefore = (u + i^2) \% m$, $m = 10$, $i = 0$ to $m-1$

$u = 7$

for $i = 0$, $= (7 + (0)^2) \% 10 = 7$ — C

$i = 1$ $= (7 + (1)^2) \% 10 = 8$ — C

$i = 2$ $= (7 + (2)^2) \% 10 = 1$ — C

$i = 3$ $= (7 + (3)^2) \% 10 = 6$ — C

$i = 4$ $= (7 + (4)^2) \% 10 = 3$ ✓

Now prepare the Hash Table

Locⁿ/Add	Key
0	13
1	9
2	
3	12
4	
5	6
6	11
7	2
8	7
9	3

Topic

Unit No. 3

Order of Element stored is ~~Hash~~ Hash Table

= { 13, 9, —, 12, —, 6, 11, 2, 7, 3 }

Double Hashing Technique

Here whenever collision is occur two hash function is used to resolve the collision

Insert K, at first free place whenever collision is occur,

Double Hashing f^n $= (u + iv) \% m$, $i = 0$ to $m-1$,

u = calculated using First Hashing f^n i.e. $H_1(K)$

V = calculated using second Hashing f^n i.e. $H_2(K)$

Example :-

key = { 3, 2, 9, 6, 11, 13, 7, 12 }

$H_1(K) = (2K+3)$, $m = 10$

$H_2(K) = (3K+1)$

$\Rightarrow$

Key	$u = H_1(K) = 2K+3 \% 10$	$V = H_2(K) = (3K+1)\%9$	loc^n	~~Probe~~ probe
3	$=(2\times3+3)\%10 = 9$		9	1
2	$=(2\times2+3)\%10 = 7$		7	1
9	$=(2\times9+3)\%10 = 1$		1	1
6	$=(2\times6+3)\%10 = 5$		5	1
11	$=(2\times11+3)\%10 = 5$	$V=(3\times11+1)\%10 = 4$	3	3
13	$=(2\times13+3)\%10 = 9$	$V=(3\times13+1)\%10 = 0$	0	—
7	$=(2\times7+3)\%10 = 7$	$V=(3\times7+1)\%10 = 2$	2	~~2~~ ~~4~~ -
12	$=(2*12+3)\%10 = 7$	$V=(3*12+1)\%10 = 7$	4	2

$$\text{Avg probe} = \frac{9}{6} = 1.5$$

Now, for key = 11 collision is occur

$\therefore$ $u = 5$, $v = 4$, $i = 0$ to $m-1$

$= (u + iv) \% m$

$i=0$, $= (5+0*4)\%10 = 5$ — c

$i=1$ $= (5+1*4)\%10 = 9$ — c

$i=2$ $= (5+2*4)\%10 = 3$ ✓

for key = 13, Collision is occur

u = 9, v = 0, i = 0 to m-1

Double Hashing $= (u + iv) \% 10$

as, $i = 0 \quad = (9 + 0*0) \% 10 = 9$

$i = 1 \quad = (9 + 1*0) \% 10 = 9$

⋮

$i = 9 \quad = (9 + 9*0) \% 10 = 9$, so we can insert 13 in the Hash Table

for key = 7, Collision is occur,

u = 7, v = 2, i = 0 to m-1, m = 10

Double Hashing $= (u + iv) \% m$

$i = 0, = (7 + 0*2) \% 10 = 7$ – c

$i = 1 \quad = (7 + 1*2) \% 10 = 9$ – c

$i = 2 \quad = (7 + 2*2) \% 10 = 1$ – c

$i = 3 \quad = (7 + 2*3) \% 10 = 3$ – c

$i = 4 \quad = (7 + 2*4) \% 10 = 5$ – c

$i = 5 \quad = (7 + 2*5) \% 10 = 7$ – c

$i = 6 \quad = (7 + 2*6) \% 10 = 9$ – c

$i = 7 \quad = (7 + 2*7) \% 10 = 1$ – c

$i = 8 \quad = (7 + 2*8) \% 10 = 3$ – c

$i = 9 \quad = (7 + 2*9) \% 10 = 5$ – c

so, it is not possible to store key = 7 in Hash Table

Now For key = 12

$u = 7$, $v = 7$, $i = 0$ to $m-1$,

$m = 10$

Double Hashing $= (u + iv) \% 10$

$i = 0$, $= (7 + 0 * 7) \% 10 = 7$ — C

$i = 1$ $= (7 + 1 * 7) \% 10 = 4$

Now Hash Table

Loch	Key
0	
1	4
2	
3	11
4	12
5	6
6	
7	2
8	
9	3

Topic | Unit No.

Order of Element in Hash Table

$= \{ -, 9, -, 11, 12, 6, -, 2, -, 3 \}$