Hashing

Tuesday, April 29, 2025 3:38 PM

1. **Why hashing?**
2. **Hashing Techniques**
   - 1.1 Division (Modulo) Method
   - 1.2 Multiplication Method
   - 1.3 Mid-Square Method
   - 1.4 Folding Method

2. Collision Resolution Techniques

- 2.1 What is a Collision?
- 2.2 Types of Collision Resolution
  - 2.2.1 Closed Hashing (Open Addressing)
    - 2.2.1.1 Linear Probing
    - 2.2.1.2 Quadratic Probing
    - 2.2.1.3 Double Hashing
  - 2.2.2 Open Hashing (Separate Chaining)
    - 2.2.2.1 Linked List Chaining
    - 2.2.2.2 Binary Tree Chaining

Why hashing?

↳ Speed of accessing the data

Linear Search:-

→ List of element → find the element SE = 96 . n = 10

max
$O(n)$ → max time

0	1	2	3	4	5	6	7	8	9
5	9	12	6	15	18	32	96	44	182
×	×	×	×	×	×	×	✓		

→ no sorting time = $O(n)$

Binary Search:-

Condition:- Sorted order

$T(n) = T(\frac{n}{2}) + 1$

se = 38

0	1	2	3	4	5	6	7	8	9
5	9	12	15	17	21	34	38	42	56

$O(\log n)$

Problem:- make elements in sorted

time = $O(\log n)$

To improve the speed of searching → better than linear & binary

hashing → $O(1)$ → fixed time.

Hashing:-

Key → data

↳ apply → hashingfunction → position

↳ division method

↳ apply → hashingfunction → position

- → division method
- → multiplication
- → folding
- ↳ mid Squares

(i) division method:- equation $\boxed{h(K) = K \bmod size}$

$10)\,17\,(1$
$\;\;\;10$
$\;\;\;(7)$

$10)\,15\,(1$
$\;\;\;10$
$\;\;\;(5)$

h(K) → hashing function

K → Key (data)

n = 10

size → size of array

elements | keys →

Keys = 5

15, 17, 21, 46, 72

K = K

	0
21	1
72	2
	3
	4
15	5
46	6
17	7
	8
	9

hash table

n = 10

index

$h(15) = 15 \% 10 = 5$

$h(17) = 17 \% 10 = 7$

$h(21) = 21 \% 10 = 1$

$h(46) = 46 \% 10 = 6$

$h(72) = 72 \% 10 = 2$

Search:- 46

$46 \% 10 = 6$

17 $\quad 17 \% 10 \Rightarrow 7$

(ii) Mid-Square method:-

$h(K) = (\text{mid of } K^2) \bmod size$

h(K) → hashing function on K

K^2 → key value × Key value

mid of K^2 → K^2 mid value

Size → Size of Array ⇒ n = 10

$12 \times 12 = 144$

data/keys = {12, 22, 32, 42, 52}

52	0
	1
32	2
	3
12	4

(i) K = 12 $\quad K^2 = 144$

data/keys = {12, 22, 32, 42, 52}

(i) $K = 12$ $K^2 = 144$

mid of K^2 = 1**4**4 ⇒ (4) mod 10 = 4
↑
mid

index

32	2
	3
12	4
	5
42	6
	7
22	8
	9

hash table

(ii) $K = 22$ $K^2 = 484$

mid of (484) = (8) % 10 = 8

10) 2 (0
0
(2)

(iii) $K = 32$ $K^2 = 1024$

mid of K^2 = (02) % 10 = 2

(iv) $K = 42$ $K^2 = 1764$.

mid of (1764) = 76 % 10 ⇒ 6

10) 76 (7
70
(6)

(v) $K = 52$ $K^2 = 2704$

mid of (2704) ⇒ 70 % 10 ⇒ '0' → index

10) 70 (7
70
(0)

(iii) Multiplication method:-

$A = 0.6180$ $A = \left\lfloor \frac{\sqrt{5}-1}{2} \right\rfloor$

$h(K) = (\underbrace{\text{size} \times K \times A}\ \text{mod}\ \text{size})$

h(K) = hash function

K = key value

$A = \frac{\sqrt{5}-1}{2} = 0.6180$

Size = Size of Array

$K = 12$

$K \times A = 12 \times 0.6180 = 7.416$

$\text{size} \times K \times A = 10 \times 7.416 \Rightarrow \lfloor 74.16 \rfloor = 74$

74 % 10 ⇒ 4

10) 74 (7
70
(4)

	0
	1
	2
	3
12	4
	5
	6
	7
	8
	9

(iv) folding method:- $\boxed{h(K) = (\Sigma \text{ Parts of } K)\ \%\ \text{size}}$

(iv) folding method:- $\boxed{h(K) = (\Sigma \text{ Parts of } K) \ \% \text{ size}}$

this method is used when your key value is large

{ 1234, 15692, 91011 }

K = 1234

12 | 34

12
34
(46) % 10 => 6

10) 46 (4
40
(6)

91011	0
	1
	2
	3
	4
	5
1234	6
	7
	8
	9

hash table

Rules,

(i) divid the given number in two digit pieces.

(ii) apply any Aritmatic operator (+)

(iii) apply mod of generated number.

with the size = 10

K = 91011

=> 9 | 10 | 11 => 9+10+11 = 30

=> 30 % 10 => 0

---

problem:- Collision

(i) division method:- h(K) = K % size

K = 12

h(K) = 12 % 10 → 2 index

K = 13

h(K) = 13 % 10 → 3 index

K = 22

h(K) = 22 % 10 → 2 index

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

22

in the above example key = 12 and key = 22 has same index.

this is called Collision

Collision Resolution Techniques

2.1 What is a Collision?

2.2 **Types of Collision Resolution**

- 2.2.1 Closed Hashing (Open Addressing)
  - 2.2.1.1 Linear Probing

- 2.2.1.2 Quadratic Probing
- 2.2.1.3 Double Hashing

- 2.2.2 Open Hashing (Separate Chaining)
  - 2.2.2.1 Linked List Chaining
  - 2.2.2.2 Binary Tree Chaining

Collision:- if two or more keys having same hash code is called collision

Keys :- K = 8, 13

$\boxed{h(K) = Key \ \% \ Size}$

$h(8) = 8 \% 5 \Rightarrow 3$

$h(13) = 13 \% 5 \Rightarrow 3$ (collision)

n = 5

	index
	0
	1
	2
8	3
	4

5)8(1
5
(3)

5)13(2
10
(3)

2.2.1.1 Linear Probing:-

if two keys having same hash value then we assign/place second key in next free index

$\boxed{h(K) = K \% size}$

Keys:- 52, 76, 94, 82, 102, 39

n = 10

A

	index
	0
	1
52	2
82	3
94	4
102	5
76	6
	7
	8
39	9

(i) $h(52) = 52 \% 10 \Rightarrow 2$ (insert)

(ii) $h(76) = 76 \% 10 \Rightarrow 6$ (insert)

(iii) $h(94) = 94 \% 10 \Rightarrow 4$ (insert)

(iv) $h(82) = 82 \% 10 \Rightarrow 2$ (collision) → if (A[2] != 0) → collision

'2' poision is collision → next immediate index = '3'

3rd slot is free so insert "82" in 3rd index

(v) $h(102) = 102 \% 10 \Rightarrow 2$ (collision).

next solt of (2) ⇒ 3 (is it free) no ⇒ next solt of (3) = 4 (is it free) no

⇒ next slot of (4) = 5 (is it free) yes → insert 102 at 5th index

(vi) $h(39) = 39 \% 10 \Rightarrow 9$ (insert).

2.2.1.2 Quaddratic probing:-

if collision occure then we use following equation (Quadratic probleng)

$$= \left(h(K) + i^2\right) \% size \quad : \quad i = 0, 1, 2, 3 \ldots$$

Keys:- 15, 24, 33, 45, 63, 94, 65

size(n) = 10

$h(K) = Key \% 10$

0 →	
1 →	
2 →	
3 →	33
4 →	24
5 →	15
6 →	45
7 →	63
8 →	
9 →	

i) $h(15) = 15 \% 10 \Rightarrow 5$ (insert -15)

ii) $h(24) = 24 \% 10 \Rightarrow 4$ (insert -24)

iii) $h(33) = 33 \% 10 = 3$ (insert -33)

iv) $h(45) = 45 \% 10 = 5$ (collision)

Quadratic probing:- $(h(K) + i^2) \% \text{size}$: $i = 0, 1, 2, 3 \ldots$

a) $h(45) = 5$, $i = 0$

$(5 + 0^2) \% 10 \Rightarrow (5+0) \% 10 \Rightarrow 5 \% 10 = 5$ (collision).

$i=1$: $(5 + 1^2) \% 10 \Rightarrow (5+1) \% 10 \Rightarrow 6 \% 10 = 6$ (inserted -45).

(v) $h(63) = 63 \% 10 = 3$ (collision)

Quadratic probing

$h(63) = 3$ $i=0$

$(3 + 0^2) \% 10 \Rightarrow (3+0) \% 10 \Rightarrow 3 \% 10 = 3$ (it is not free - Collision).

$\boxed{\begin{array}{l} i = 3 \quad \text{Rough} \\ (3 + 3^2) \% 10 \\ \Rightarrow (3+9) \% 10 \\ = (12) \% 10 = 2 \end{array}}$

$i=1$

$(3 + 1^2) \% 10 \Rightarrow (3+1) \% 10 \Rightarrow 4 \% 10 = 4$ (it is not free - Collision)

$i=2$

$(3 + 2^2) \% 10 \Rightarrow (3+4) \% 10 \Rightarrow 7 \% 10 = 7$ (it is free) → insert 63

n = 20
19

2, 3, 5, 7 ↑ prime
n = 10

n = 100
97

2.2.1.3:- Double hasing:-

$h(K) = (h_1(K) + i * h_2(K)) \% m$

$i = 0, 1, 2, 3, 4 \ldots$

$h_2(K) = \text{prime} - (\text{key} \% \text{prime})$

Keys:- 15, 24, 33, 45, 63, 94, 65

0 →	63
1 →	
2 →	
3 →	33
4 →	24
5 →	15
6 →	
7 →	
8 →	
9 →	45

$\left.\begin{array}{l} h(15) = 15 \% 10 = 5 \\ h(24) = 24 \% 10 = 4 \\ h(33) = 33 \% 10 = 3 \end{array}\right\}$ inserted

$h(45) = 45 \% 10 = 5$ (Collision).

→ Double hashing:- $h_2(K) = \text{prime} - (\text{key} \% \text{prime})$

prime = 7

$h_2(K) = 7 - (45 \% 7) = 7 - (3) = 4$

$7 \overline{)45} (6$
42
(3)

$h_2(K) = 7 - (45 \% 7) = 7 - (3) = \underline{4}$

$i = 0$ $\quad h(45) = (h_1(45) + 0 * h_2(45)) \% 10$

$= (\underline{5} + 0 * 4) \% 10 \Rightarrow (5) \% 10 = \underline{5}$ (Collision)

$(i = 1)$ $\quad h(45) = (h_1(45) + 1 * h_2(45)) \% 10$

$= (5 + 1 * 4) \% 10$

$= (5 + 4) \% 10 \Rightarrow 9 \% 10 = \underline{9}$ (insert - 45)

$10 \overline{) 9} (0$
0
$\underline{(9)}$

$h(63) = 63 \% 10 \Rightarrow \underline{3}$ (Collision)

Double hashing:

$h_2(63) = prime - (key \% prime)$

$= \underline{7} - (63 \% 7) \Rightarrow 7 - (0) = \underline{7}$

$7 \overline{) 63} (9$
63
$\underline{(0)}$

$h(63) = (h_1(63) + i * h_2(63)) \% size$

$i = 0$ $\quad h(63) = (3 + 0 * 7) \% 10$

$= 3 \% 10 \Rightarrow \underline{3} = h(63)$ (collision)

$10 \overline{) 10} (1$
10
$\underline{(0)}$

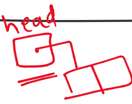
$i = 1$ $\quad h(63) = (h_1(63) + 1 * h_2(63)) \% size$

$= (3 + 1 * 7) \% 10$

$\Rightarrow (10 \% 10) = \underline{0}$ (insert 63 at 0)

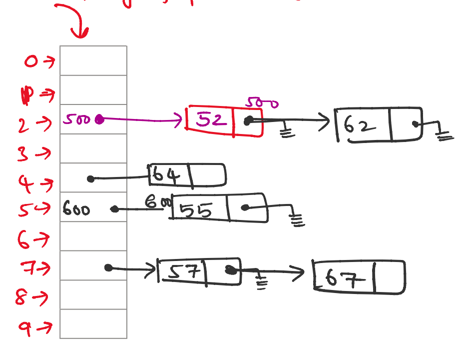
---

head



* Linked list Chaining:

Array of pointer (it holds address)



Keys: 52, 55, 57, 62, 64, 67, 71, 75, 77

(i) $h(52) = 52 \% 10 \Rightarrow$ '2'

(ii) $h(55) = 55 \% 10 \Rightarrow 5$

(iii) $h(57) = 57 \% 10 \Rightarrow 7$

(iv) $h(62) = 62 \% 10 \Rightarrow 2$ (Collision) → new

(v) $h(64) = 64 \% 10 = 4$

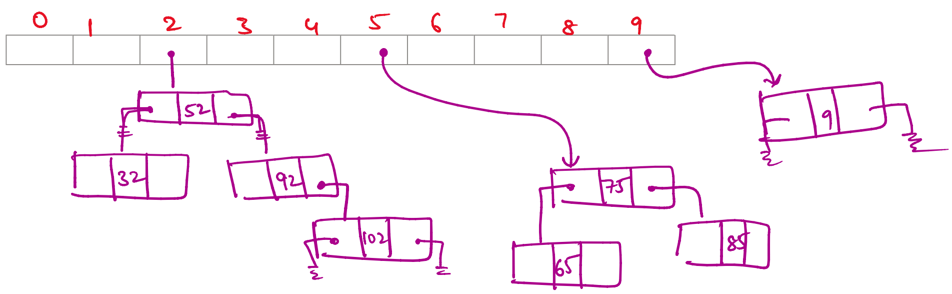
(iv) $h(62) = 62 \% 10 \Rightarrow 2$ (Collision) → new

(v) $h(64) = 64 \% 10 = \underline{4}$

(vi) $h(67) = 67 \% 10 = \underline{7}$

Binary Tree Chaining :

$n = 10$ Array of Pointers →



Keys : 52, 32, 92, 102, 75, 65, 85, 99

$h(52) = 52 \% 10 = 2$

$h(32) = 32 \% 10 = 2$

$h(92) = 92 \% 10 = 2$

$h(102) = 102 \% 10 = 2$

$h(75) = 75 \% 10 = 5$

$h(65) = 65 \% 10 = 5$

$h(85) = 85 \% 10 = 5$

$h(99) = 99 \% 10 = 9$