Data Structures and its Applications

V R BADRI PRASAD
Department of Computer Science & Engineering

DATA STRUCTURES AND ITS APPLICATIONS

Introduction to Hashing :

- Hash Function
- Hash Table
- Creation of Hash Table

V R BADRI PRASAD

Department of Computer Science & Engineering

Introduction to Hashing

- Sequential search algorithm takes lot of time for searching o(n).
- BST improves performance by log(n).
  - To achieve this speed, BST should be balanced.

Consider an employee database of 10000 records.

Using

- Linked list would take O(n) time.
- Using Balanced BST would take O(log n) time.
- However, using arrays, would take O(1) time will lead to a lot of space wasted.
- Is there a way to get the data retrieved with O(1) time without memory being wasted?
- The solution is HASHING.

Data Structures and its Applications
Hashing – Hash Function and Hash Table

- Implementing Dictionaries
- Takes equal time for operation
- Efficient techniques for retrieval of data would be one that takes less number of comparisions.
- A **hash table**, or a **hash map**, is a data structure that associates keys (names) with values (attributes).
- Use hash function to map keys to hash tables.
- Key is stored at a memory location , the address of the location is computed using hash function.

Example:

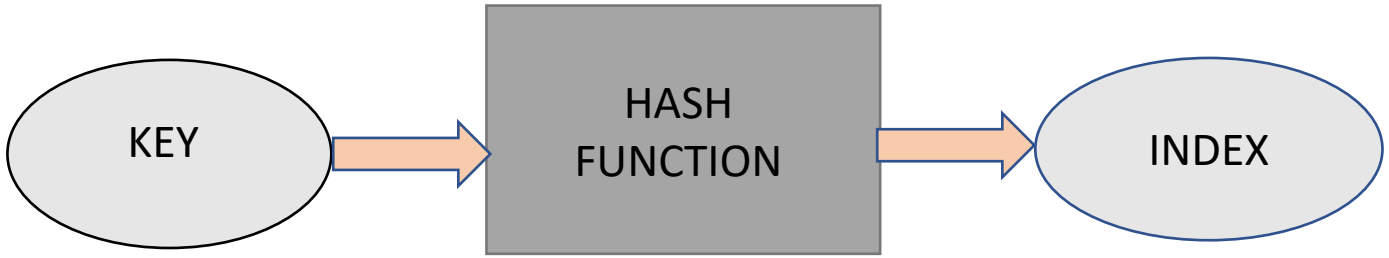
• Consider a key 496000. Suppose the hash table has 10 memory locations, then the key is stored at location which has an address computed using hash function key mod 10.

Address(index) is : 496005 mod 10 = 5.

The data 496000 is stored at location with index five.

Hashing – Hash Function and Hash Table

- I A good hash function is one that distributes keys evenly among all slots / index (locations).
- Design of a hash function is an art more than science.



Hash Table

Index / hash	DATA
0	15
1	46
2	72
3	18
4	34

- Consider key elements as 34, 46, 72, 15, 18, 26, 93
- Hash function is **key mod 5.**
- Index value for the keys are generated using the given hash function

- 34 mod 5 = 4, 34 is stored at index 4.
- 46 mod 5 = 1, 46 is stored at index 1.
- 72 mod 5 = 2 , 72 is stored at index 2.
- 15 mod 5 = 0, 15 is stored at index 0.
- 18 mod 5 = 3, 18 is stored at index 3.

•This technique is called **closed hashing**

Hashing: creation of Hash Table , Collision.

Hash Table	
Index	**DATA**
0	15
1	46
2	72
3	18
4	34

- Consider key elements as 34, 46, 72, 15, 18, 26, 93
- Hash function is **key mod 5.**
- Index value for the keys are generated using the given hash function

  - 34 mod 5 = 4, 34 is stored at index 4.
  - 46 mod 5 = 1, 46 is stored at index 1.
  - 72 mod 5 = 2 , 72 is stored at index 2.
  - 15 mod 5 = 0, 15 is stored at index 0.
  - 18 mod 5 = 3, 18 is stored at index 3.
  - **26 mod 5 =1, but location 1 is already occupied.**

    Hence results in clash / collision.

Also, the capacity of the hash table is full.

Hence, 26 cannot be stored in the hash table.

Hash Table : Resolving Collision

It is true for the next data item 93 as location with index 3 is also occupied.

Which results in clash.

The problem can be resolved by

- Increasing the Memory Capacity.
- Overcoming Collision using
  - Open Addressing / Separate Chaining
  - Closed Addressing :
    - Linear Probing
    - Quadratic Probing
    - Double Hashing

Hashing – Open Addressing / Separate Chaining

Initially Hash Table contains all 'NULL' values in the address field of the hash table.

- Consider key elements as 34, 46, 72, 15, 18
- Hash function is **key mod 5.**

Hash Table	
Index	**address**
0	NULL
1	NULL
2	NULL
3	NULL
4	NULL

- 34 mod 5 = 4, 34 is stored at index 4.
- 46 mod 5 = 1, 46 is stored at index 1.
- 72 mod 5 = 2 , 72 is stored at index 2.
- 15 mod 5 = 0, 15 is stored at index 0.
- 18 mod 5 = 3, 18 is stored at index 3.

Hashing – Open Addressing / Separate Chaining

Initially Hash Table contains all 'NULL' values in the address field of the hash table.

- Consider key elements as 34, 46, 72, 15, 18
- Hash function is **key mod 5.**

Hash Table

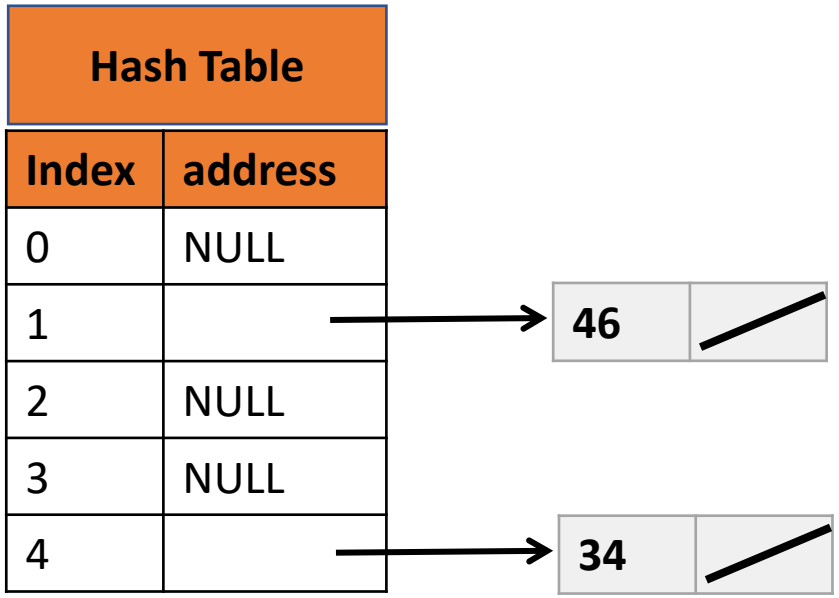
Index	address
0	NULL
1	NULL
2	NULL
3	NULL
4	

- **34 mod 5 = 4, 34 is stored at index 4.**
- 46 mod 5 = 1, 46 is stored at index 1.
- 72 mod 5 = 2 , 72 is stored at index 2.
- 15 mod 5 = 0, 15 is stored at index 0.
- 18 mod 5 = 3, 18 is stored at index 3.

Hashing – Open Addressing / Separate Chaining

Initially Hash Table contains all 'NULL' values in the address field of the hash table.

- Consider key elements as 34, 46, 72, 15, 18
- Hash function is **key mod 5.**

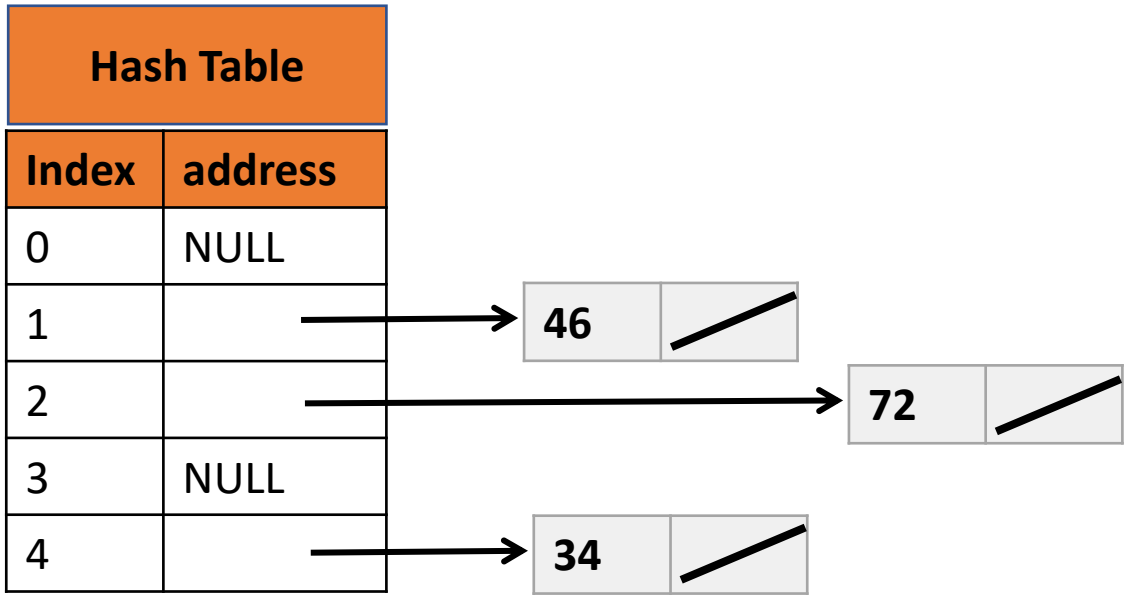


- 34 mod 5 = 4, 34 is stored at index 4.
- **46 mod 5 = 1, 46 is stored at index 1.**
- 72 mod 5 = 2 , 72 is stored at index 2.
- 15 mod 5 = 0, 15 is stored at index 0.
- 18 mod 5 = 3, 18 is stored at index 3.

Hashing – Open Addressing / Separate Chaining

Initially Hash Table contains all 'NULL' values in the address field of the hash table.

- Consider key elements as 34, 46, 72, 15, 18
- Hash function is **key mod 5.**

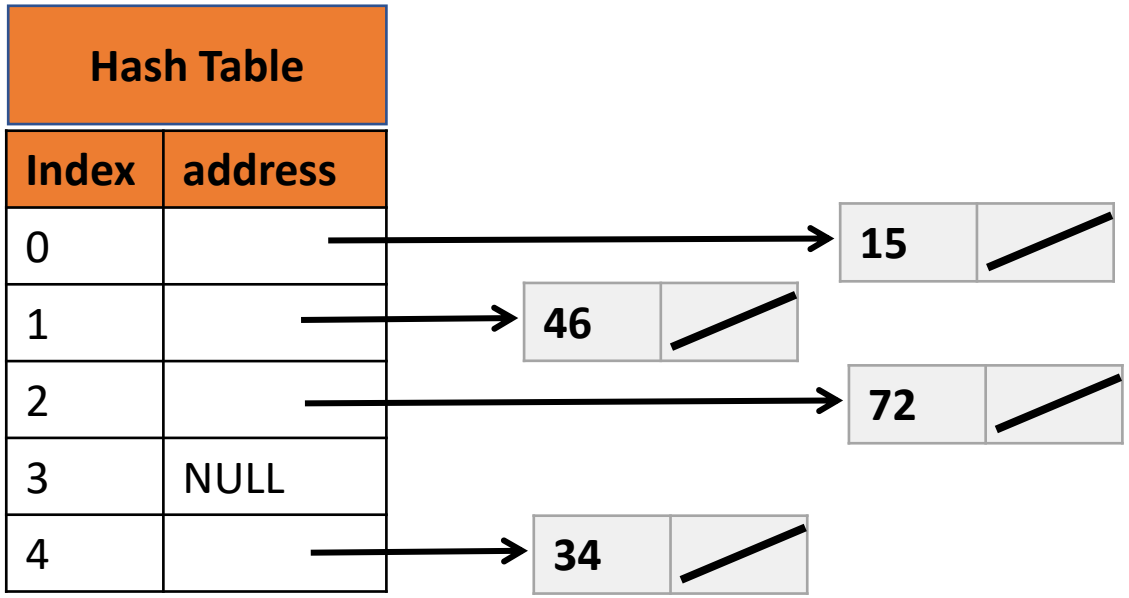


- 34 mod 5 = 4, 34 is stored at index 4.
- 46 mod 5 = 1, 46 is stored at index 1.
- **72 mod 5 = 2 , 72 is stored at index 2.**
- 15 mod 5 = 0, 15 is stored at index 0.
- 18 mod 5 = 3, 18 is stored at index 3.

Hashing – Open Addressing / Separate Chaining

Initially Hash Table contains all 'NULL' values in the address field of the hash table.

- Consider key elements as 34, 46, 72, 15, 18
- Hash function is **key mod 5.**

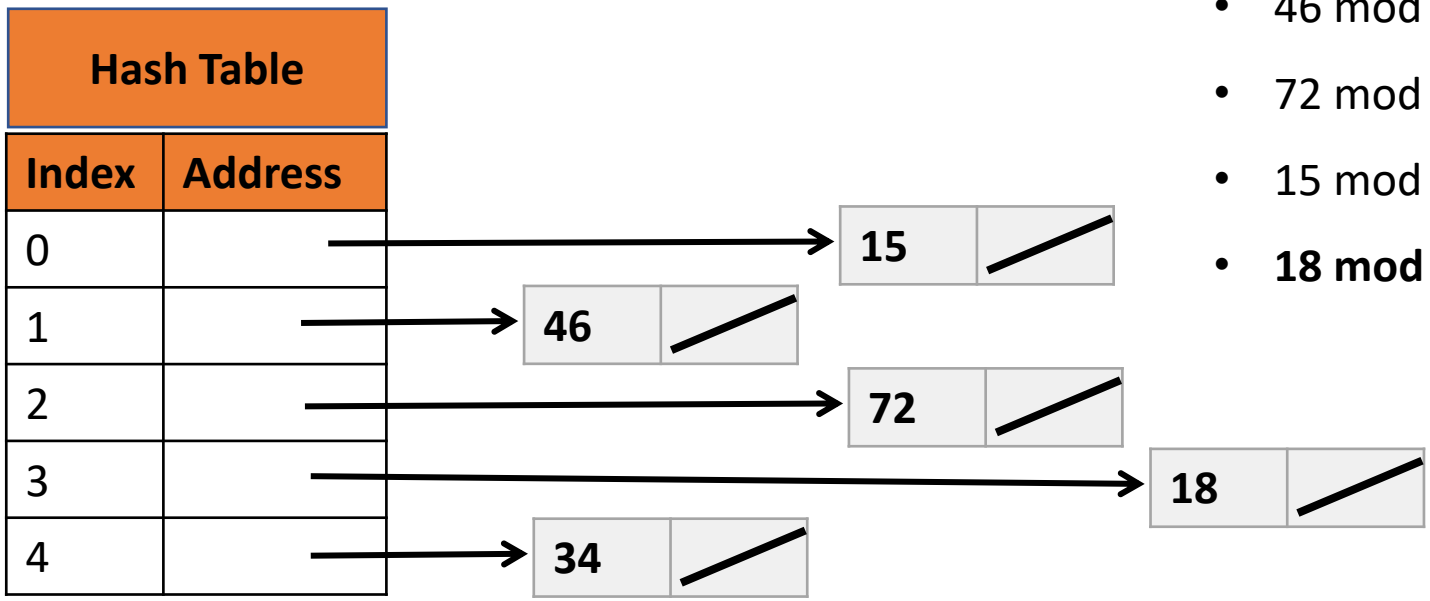


- 34 mod 5 = 4, 34 is stored at index 4.
- 46 mod 5 = 1, 46 is stored at index 1.
- 72 mod 5 = 2 , 72 is stored at index 2.
- **15 mod 5 = 0, 15 is stored at index 0.**
- 18 mod 5 = 3, 18 is stored at index 3.

Hashing – Open Addressing / Separate Chaining

Initially Hash Table contains all 'NULL' values in the address field of the hash table.

- Consider key elements as 34, 46, 72, 15, 18
- Hash function is **key mod 5.**



- 34 mod 5 = 4, 34 is stored at index 4.
- 46 mod 5 = 1, 46 is stored at index 1.
- 72 mod 5 = 2 , 72 is stored at index 2.
- 15 mod 5 = 0, 15 is stored at index 0.
- **18 mod 5 = 3, 18 is stored at index 3.**

THANK YOU

V R BADRI PRASAD

Department of Computer Science & Engineering

badriprasad@pes.edu