



CS-2001

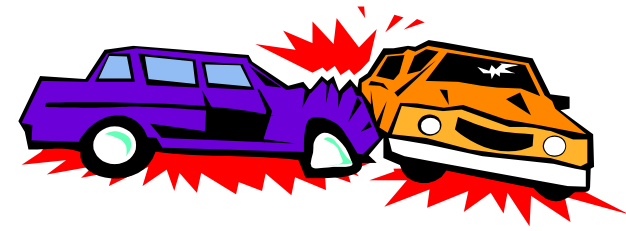
DATA STRUCTURE

Dr. Hashim Yasin

**National University of Computer
and Emerging Sciences,
Faisalabad, Pakistan.**

HASHING

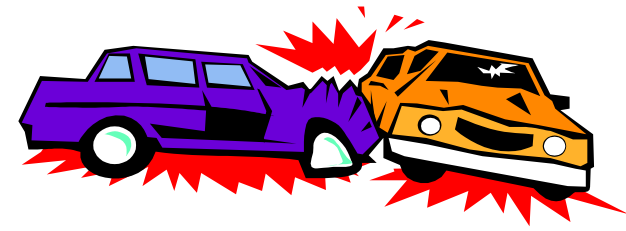
Collision



3

- The condition resulting when two or more keys produce the same hash location.
- *A good hash function minimizes collisions* by spreading the elements uniformly throughout the array.

Collision



4

- Collision handling techniques
 - Linear Probing
 - Rehashing
 - Double Hashing
 - Quadratic Probing
 - Random Probing
 - Buckets
 - Chaining

Collision Resolution Techniques

5

- There are two broad ways of collision resolution:

1. Open Addressing: Array-based implementation.

- (i) Linear probing (linear search)
- (ii) Quadratic probing (nonlinear search)
- (iii) Double hashing (uses two hash functions)

2. Separate Chaining: A linked list implementation

BUCKET

Bucket

7

- A collection of elements associated with a particular hash location
- Handle collision by allowing multiple-element keys to hash to the same location
- A solution is to let each computed hash location contain slots for multiple elements
- Each of these multi-element locations is called a bucket

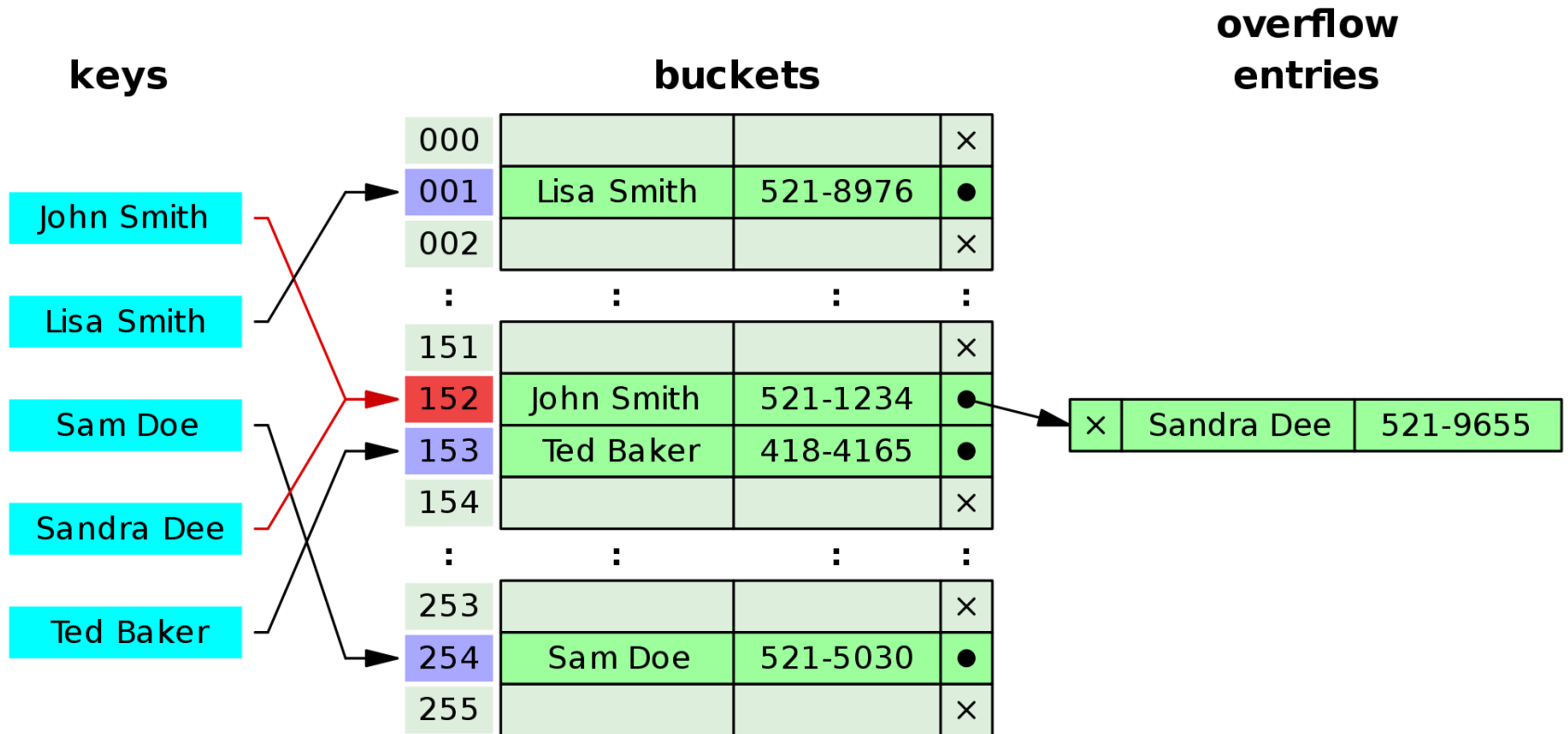
Bucket

8

- ❑ Slots are grouped into buckets
- ❑ The hash function transforms the key into a bucket number
- ❑ Each bucket contains B slots, and no collision occurs until the bucket is full.
- ❑ At that point you need to apply a collision processing strategy to find another bucket

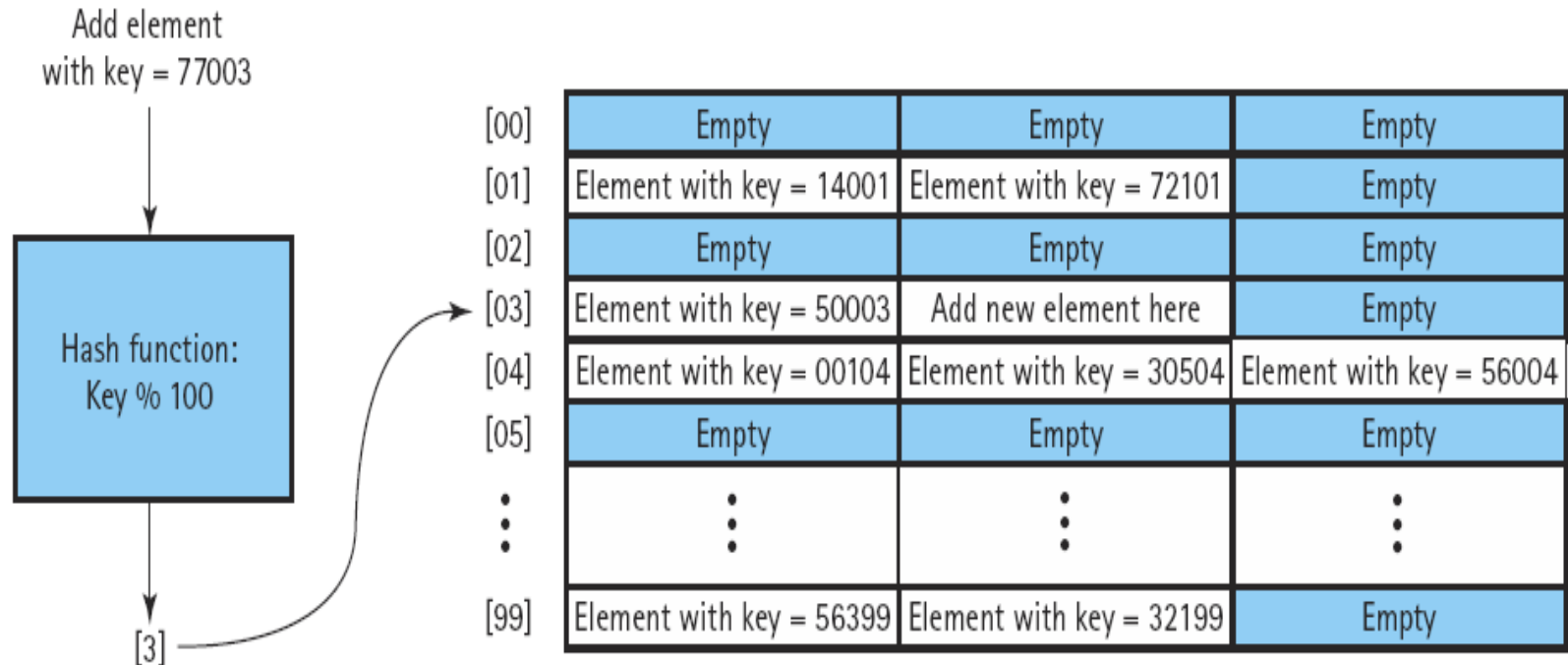
Bucket-Example

9



Bucket

10



- When the bucket becomes full, we must again deal with the problem of handling collision

CHAIN

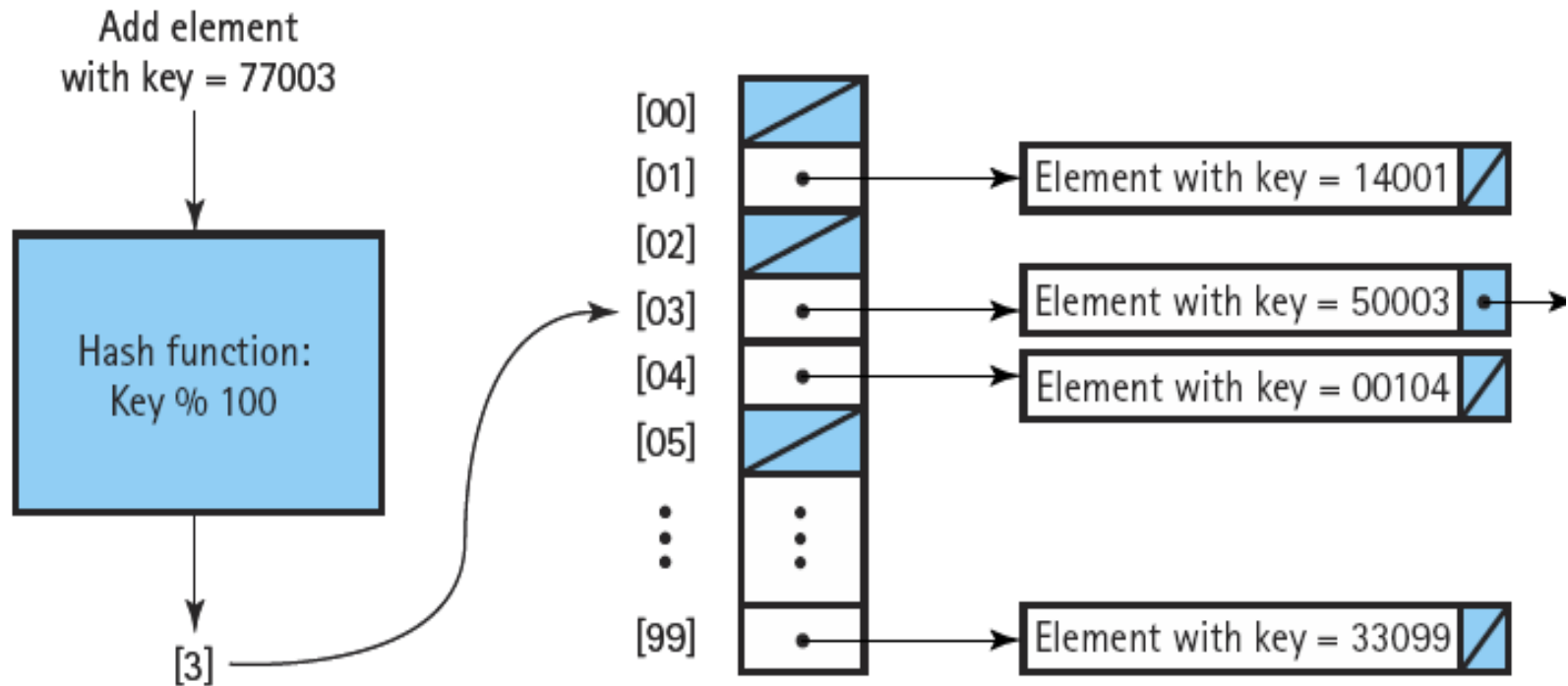
Chain

12

- A linked list of elements that share the same hash location
- Use the hash value not as the actual location of the element, but rather as the *index into an array of pointer*
- *Each pointer accesses a chain of elements that share the same hash location*
- Good when deleting an element from the linked list

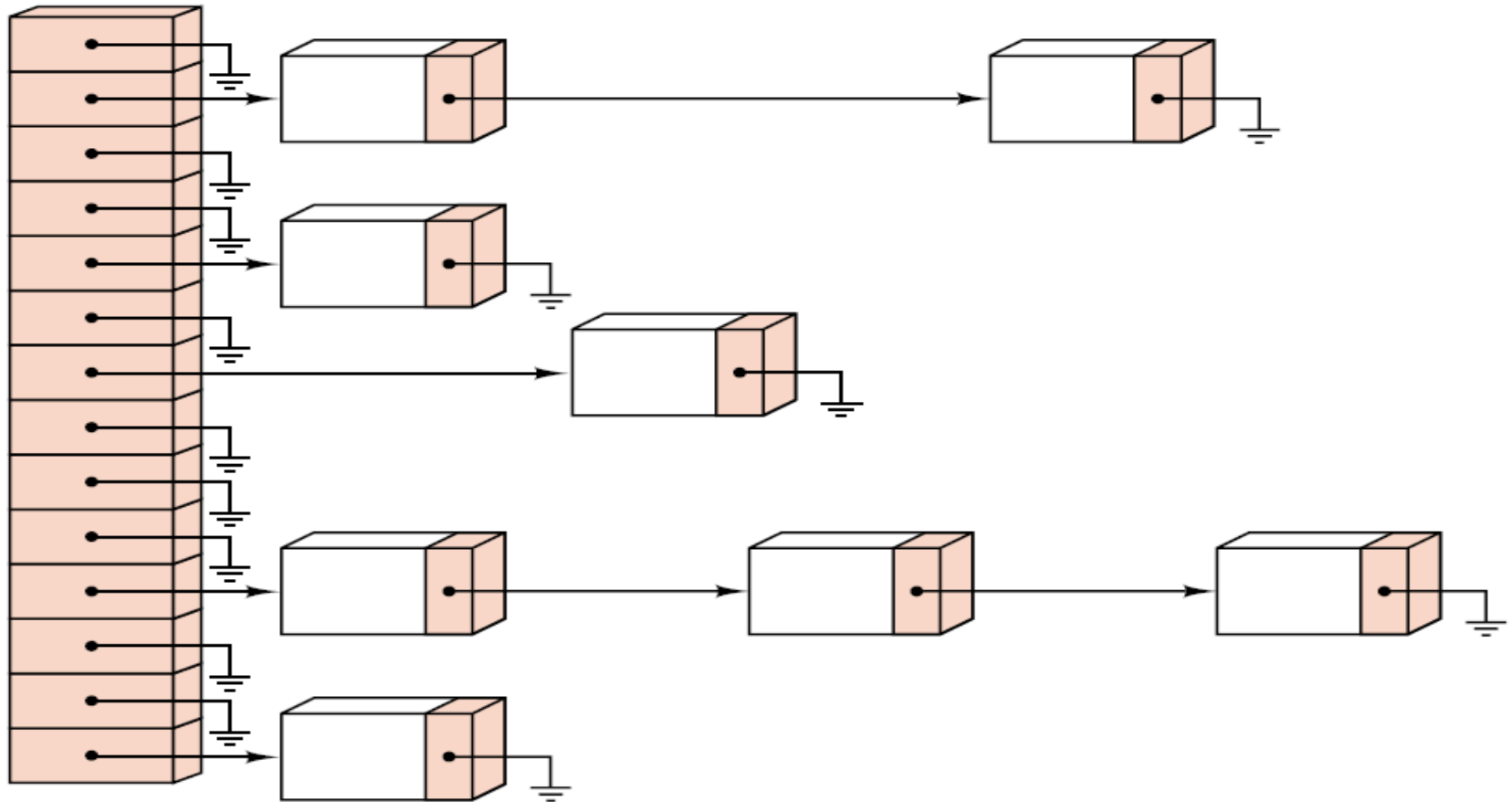
Chain

13



Chain

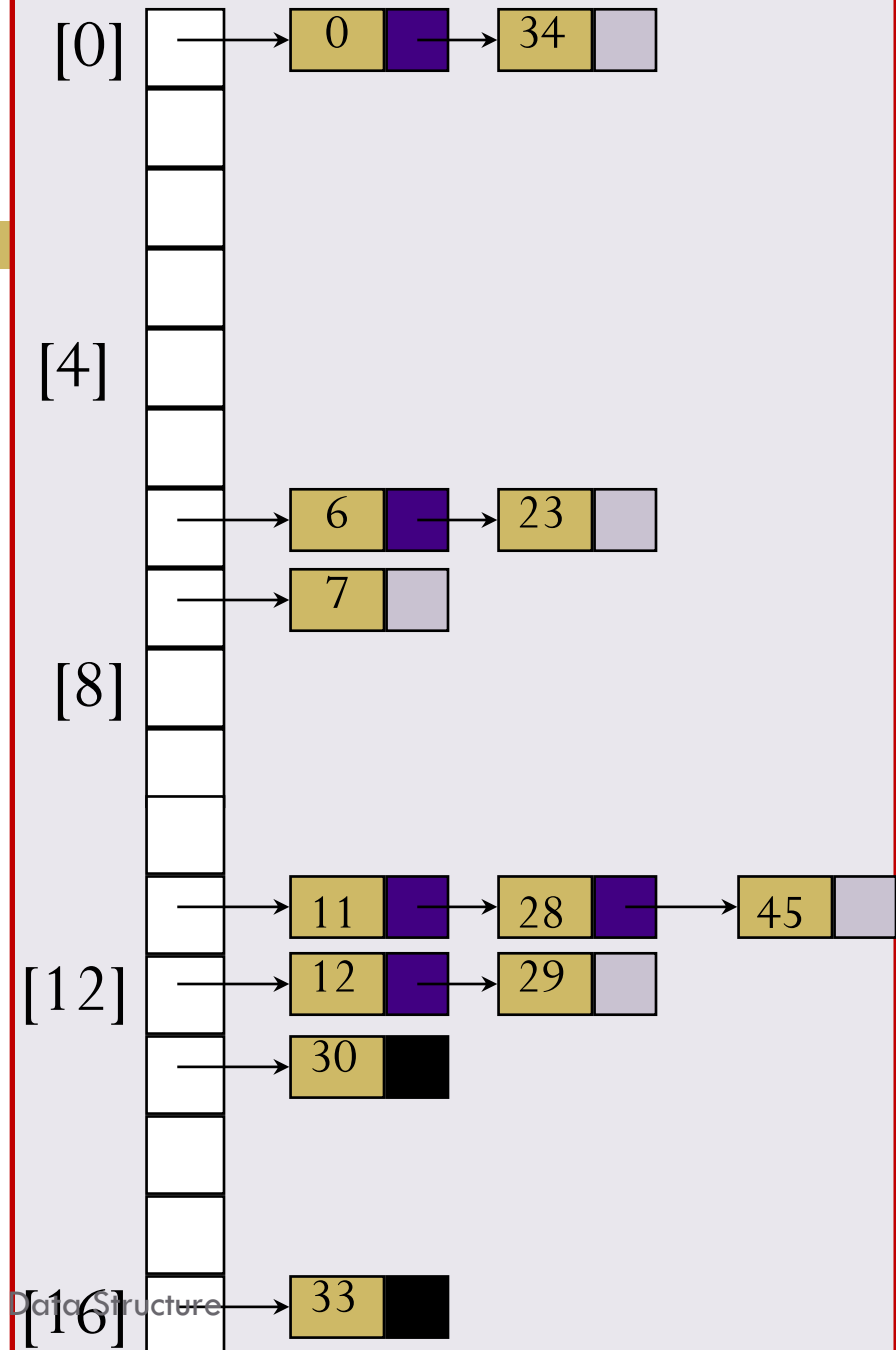
14



Chain

15

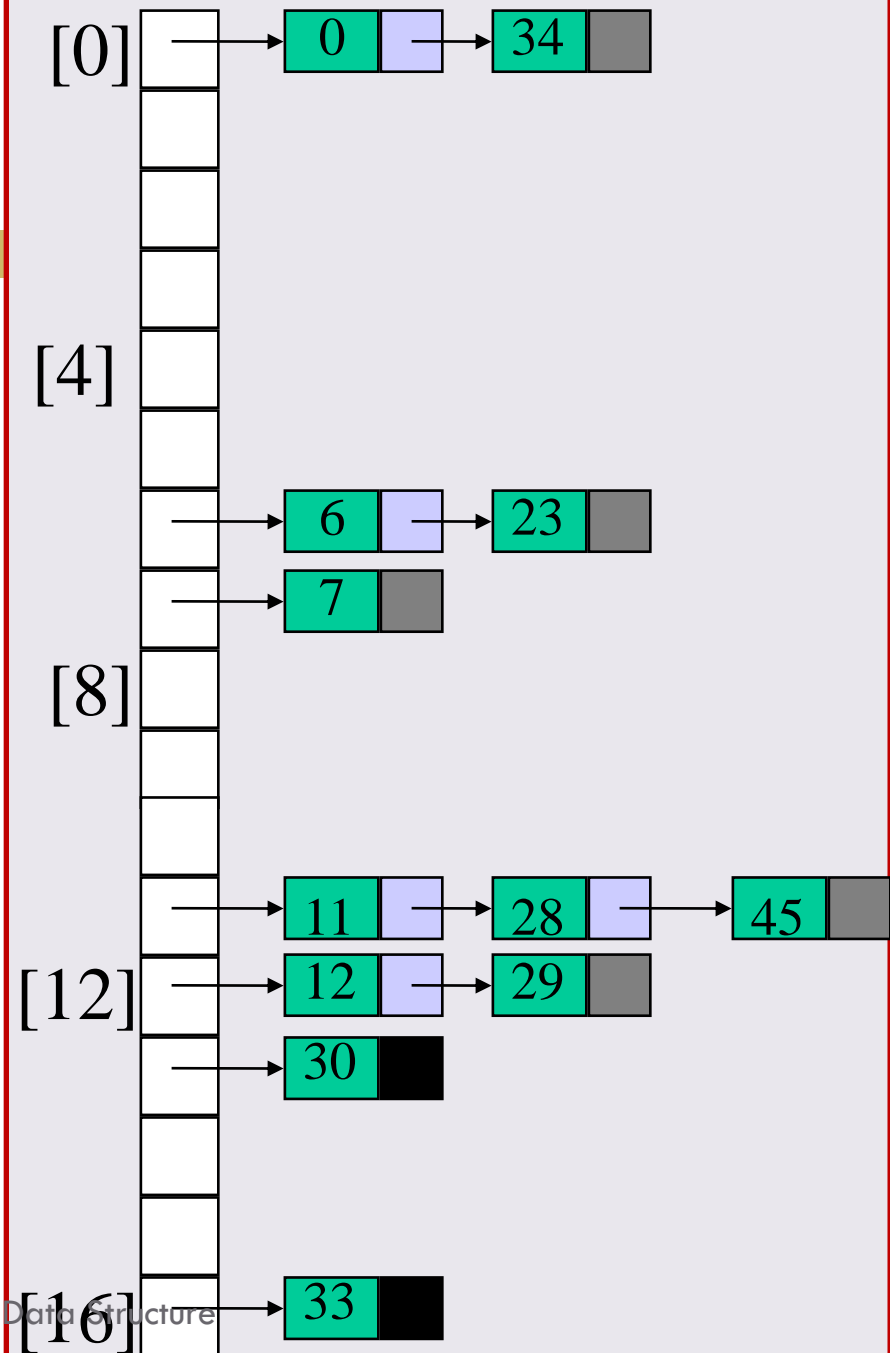
- Use linked list
 - ▣ to connect the identifiers with the same hash value and
 - ▣ to increase the capacity of a bucket.



Sorted Chains

16

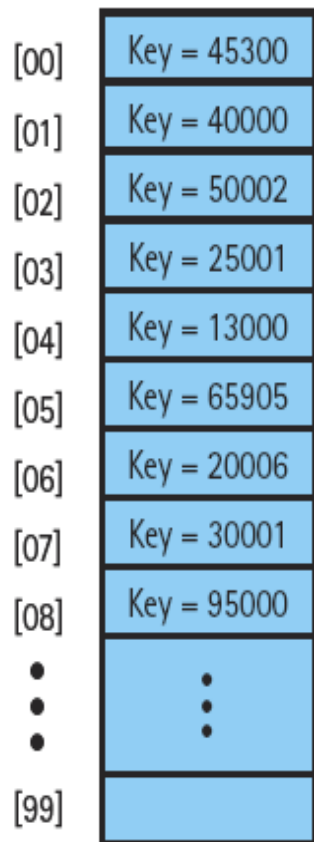
- Put in pairs whose keys are 6, 12, 34, 29, 28, 11, 23, 7, 0, 33, 30, 45
- Bucket = key % 17.



Comparison – Linear Probing & Chaining

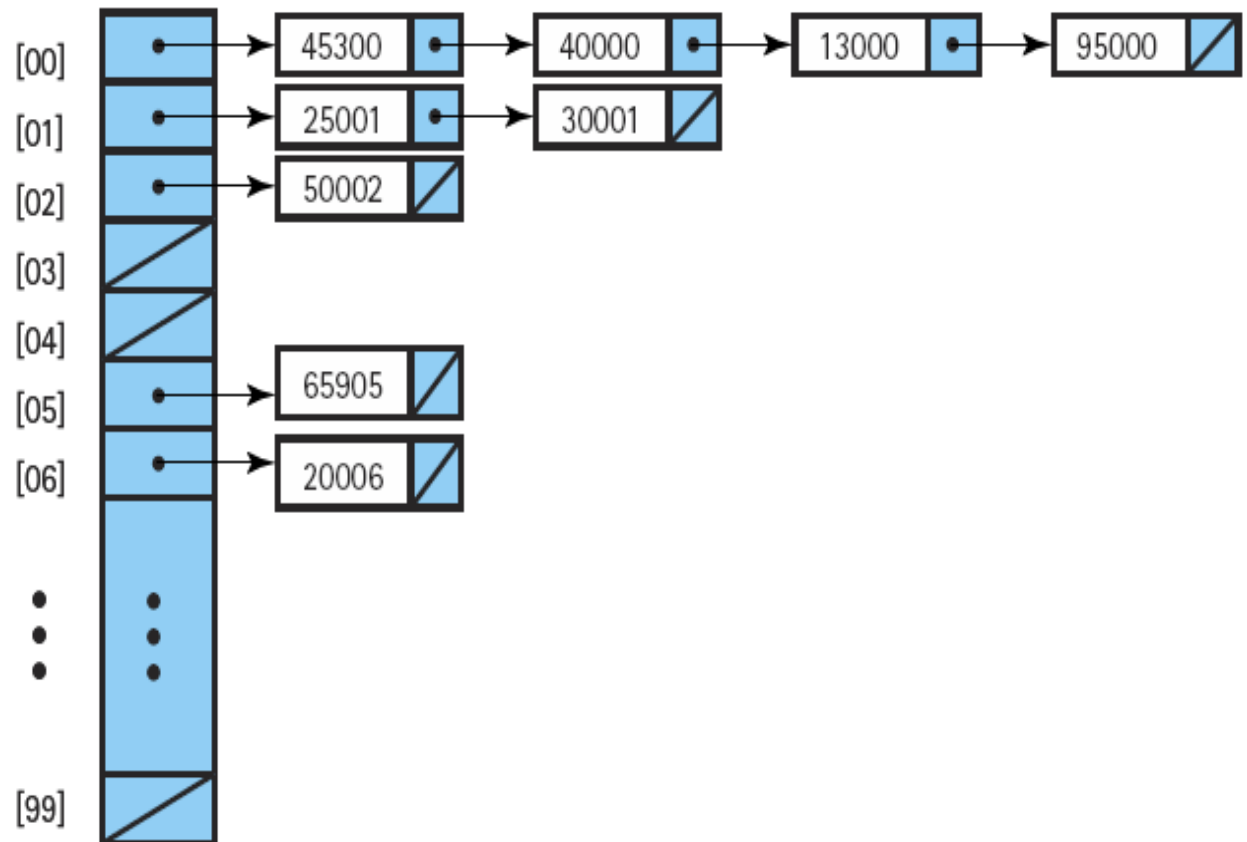
17

(a) Linear Probing



Dr. Hashim Yasin

(b) Chaining



...

CS-2001 Data Structure

SEPARATE CHAINING

Separate Chaining

19

- Retrieval of an item, **r**, with hash address, **i**, is simply retrieval from the linked list at position **i**.
- Deletion of an item, **r**, with hash address, **i**, is simply deleting **r** from the linked list at position **i**.

Separate Chaining

20

Example: Load the keys **23, 13, 21, 14, 7, 8, and 15** , in this order, in a hash table of size **7** using separate chaining with the hash function: **$h(\text{key}) = \text{key} \% 7$**

$$h(23) = 23 \% 7 = 2$$

$$h(13) = 13 \% 7 = 6$$

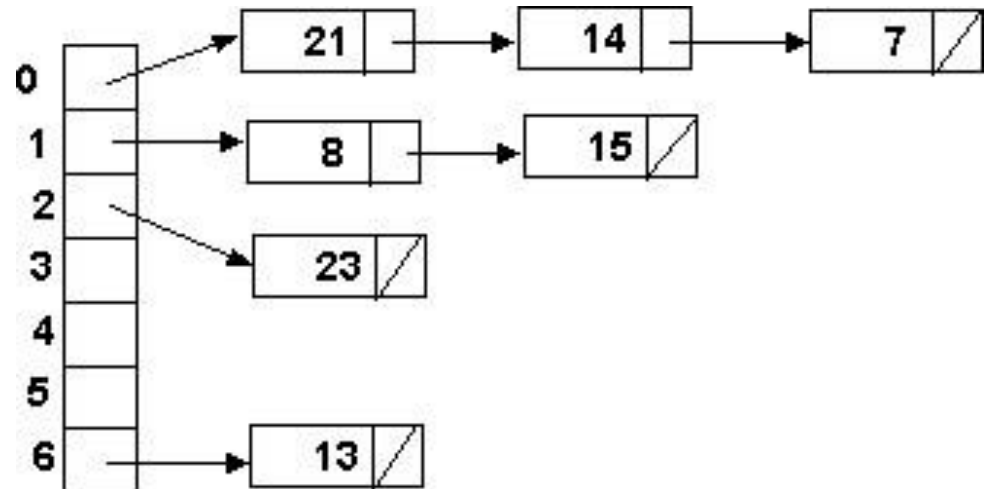
$$h(21) = 21 \% 7 = 0$$

$$h(14) = 14 \% 7 = 0 \quad \text{collision}$$

$$h(7) = 7 \% 7 = 0 \quad \text{collision}$$

$$h(8) = 8 \% 7 = 1$$

$$h(15) = 15 \% 7 = 1 \quad \text{collision}$$



Designing a good Hash Function

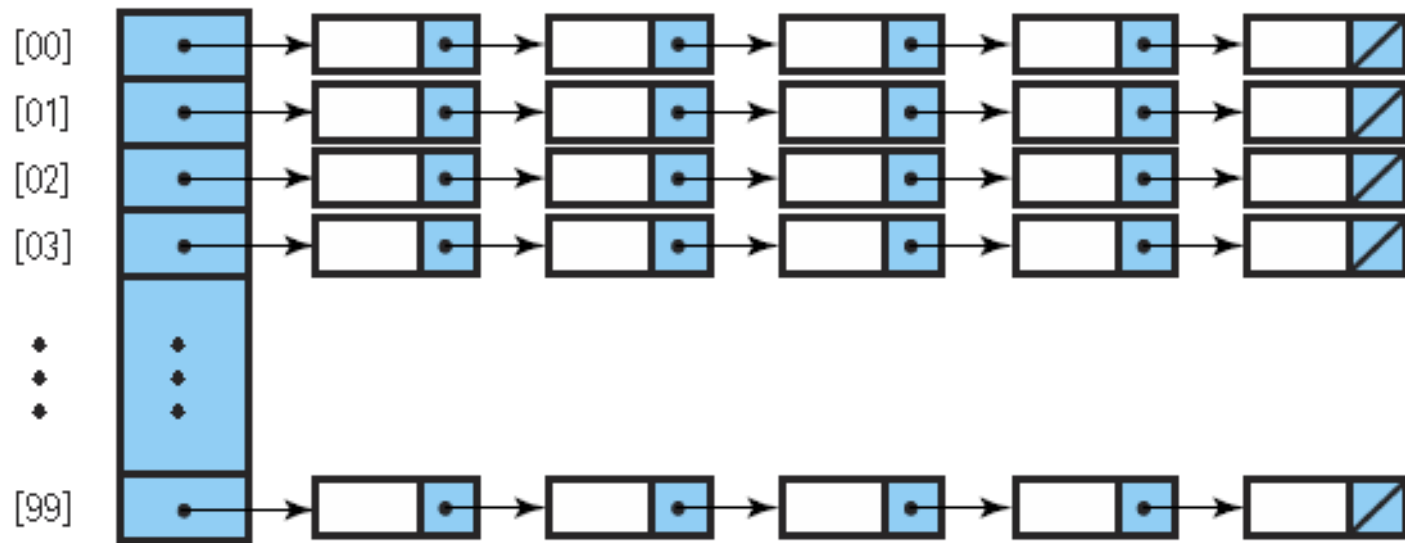
21

- A good hash function minimize the collisions
- One Solution
 - Use a data structure that has more space for keys
- Another Solution
 - Design hash function to minimize the collisions
 - Produce unique keys as much as possible
- To avoid collision causing worst case need to know statistical distribution of keys.

Choosing Good Hash Function

22

(a) The plan



Average 5 records/chain

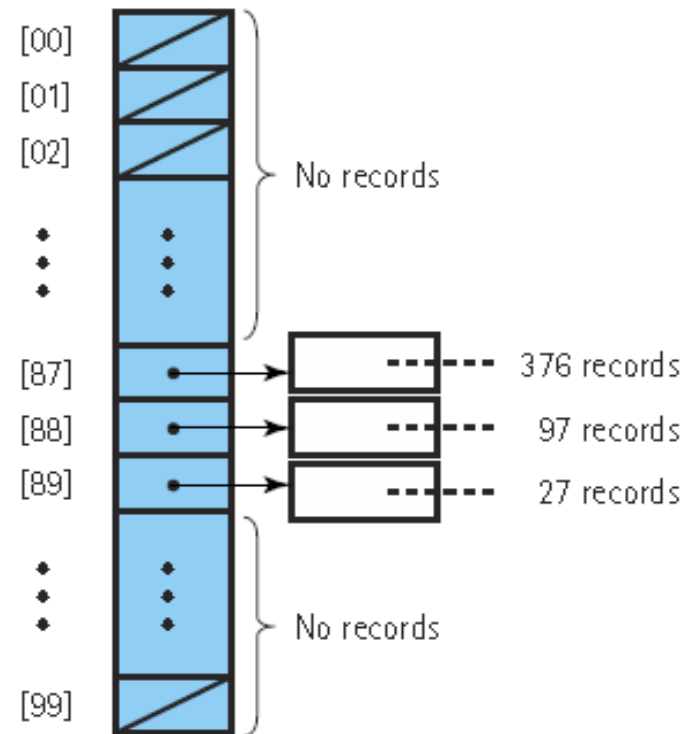
5 records \times 100 chains = 500 employees

Expected search — $O(5)$

Contd..

23

(b) The reality



376 employees hired in 1987

97 employees hired in 1988

27 employees hired in 1989

500 employees

Actual search of 80

Contd...

24

- While choosing a good hash function consider the following two things:
- First, consider the efficiency of calculating the function.
 - ▣ Even if a hash function always produces unique values, it is not a good hash function if *it takes longer to calculate the hash value than to search half the list.*

Contd..

25

- **Second**, *consider the program time.*
 - ▣ An extremely exotic function that somehow produces unique hash values for all of the known key values may fail if the domain of possible key values changes in a later modification.
 - ▣ The programmer who has to modify the program may then waste a lot of time trying to find another hash function that is equally clever.

Separate Chaining versus Open-addressing

26

□ Open Addressing

- ▣ All items are stored in the hash table itself.
- ▣ In addition to the cell data (if any), each cell keeps one of the three states: EMPTY, OCCUPIED, DELETED.
- ▣ While inserting, if a collision occurs, alternative cells are tried until an empty cell is found.

Separate Chaining versus Open-addressing

27

Disadvantages of Separate Chaining:

- It requires the implementation of a separate data structure for chains, and code to manage it.
- The main cost of chaining is the extra space required for the linked lists.
- For some languages, creating new nodes (for linked lists) is expensive and slows down the system.

References

28

- Nell Dale – Chapter 10.
- <http://www.cplusplus.com/doc/tutorial/templates/>
- Robert Lafore, Chapter 14, Page 681

