MIS 301 RELATIONAL DATABASE MANAGEMENT SYSTEM

DATABASE MANAGEMENT SYSTEM

Concepts of hashing (static, dynamic)

Lecture 30&31

HASHING

- It is tough to search all the index values through all its levels in indexing if the data file is huge.
- Hashing method is used to index and retrieve items in a database using the shorter hashed key instead of using its original value.
- Hashing is an ideal method to calculate the direct location of a data record on the disk without using index structure.
- A hash function, is a mapping function which maps all the set of search keys to the address where actual records are placed.
- The result of a hash function is known as a **hash value** or simply, a **hash**.
- Two keys can generate the same hash. This phenomenon is known as a **collision**. This leads**.** to **bucket-overflow**

HASHING

- A **hash table** stores key/value pairs in the form of a list where any element can be accessed using its index.
- **Data buckets** are memory locations where the records are stored.
- Hashing is used in data encryption. Passwords can be stored in the form of their hashes.

HASHING

- Bucket is a unit of storage. A bucket stores one complete disk block, which in turn can store one or more records.
- A hash function is a mapping function that maps the set of search-keys to the address where actual records are placed.

HASHING METHODS

- **Static Hashing**
  - The resultant data bucket address will always remain the same.
  - Therefore the number of data buckets in memory always remains constant.
- **Dynamic Hashing**
  - data buckets are added and removed dynamically and on demand.
  - the hash function helps creation of a large number of values.

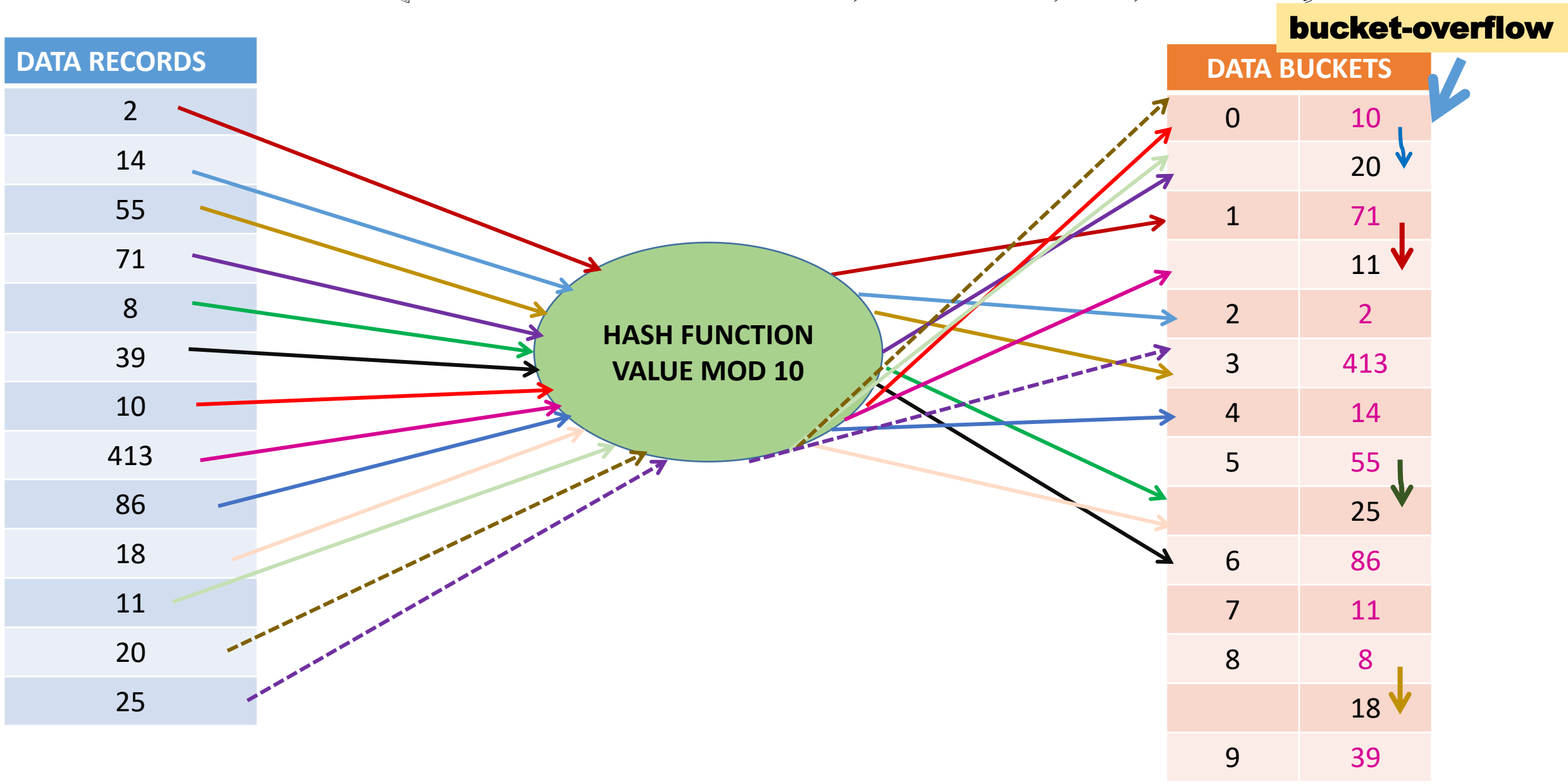
REMOVAL OF HASHING COLLISION

- **Rehashing**: This method, invokes a secondary hash function, which is applied continuously until an empty slot is found, where a record should be placed.
- **Chaining**: Chaining method builds a Linked list of items whose key hashes to the same value. This method requires an extra link field to each table position.

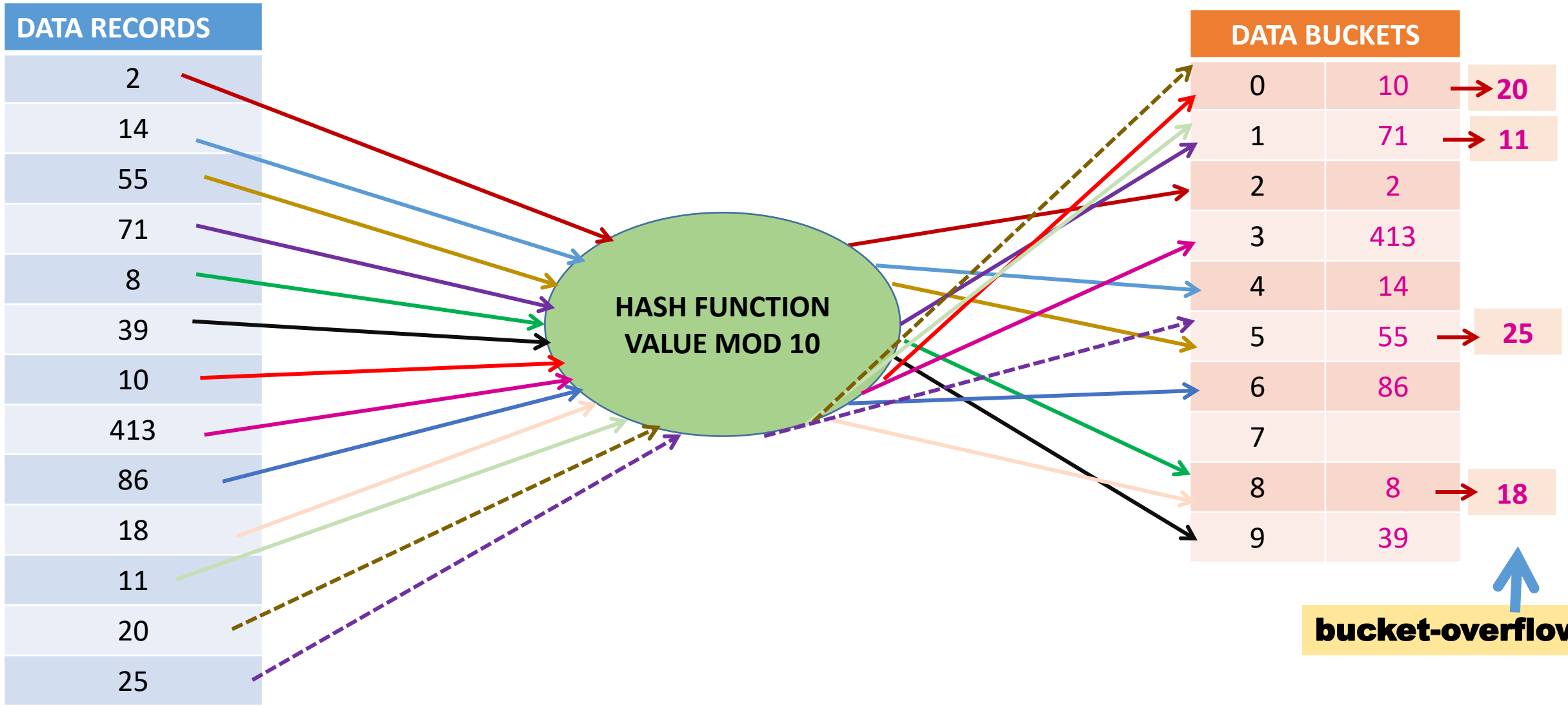
STATIC HASHING

- The number of data buckets in memory remains constant throughout.
- The address generated for the same data using the hash function will lead to the same bucket always.
- To overcome bucket overflow two methods are used namely **open hashing** and **close hashing.**
- When a hash function generates an address at which data is already stored, then the next bucket is allocated to it in open hashing.
- When buckets are full, then a new data bucket is allocated for the same hash result and is linked after the previous one in close hashing.

STATIC HASHING(OPEN HASHING OR LINEAR PROBING)



STATIC HASHING(CLOSE HASHING OR OVERFLOW CHAINING)



DYNAMIC HASHING

- Dynamic hashing overcomes the disadvantage of bucket overflow.
- Data buckets increase or decrease as records are added or removed.
- Insertion and deletion operations do not lead to a fall in performance.
- As data increases, the size of the memory is increased to accommodate the data.
- Memory is well utilized as it grows and shrinks with the data leaving no unused memory.
- Addresses of data are maintained in the **bucket address table**.
- Every hash index has a depth value to signify how many bits are used for computing a hash function.
- When all the buckets are full – then the depth value is increased linearly and twice the buckets are allocated.
- When data are not ordered but discrete and random, hash performs the best.

DYNAMIC HASHING

DATA RECORDS	HASH FUNCTION	BUCKET NO.
2	10	B2
14	1110	B2
55	110111	B3
71	1000111	B0
8	1000	B0
39	100111	B3
10	1010	B2
413	110011101	B1
86	1010110	B2
18	10010	B2
11	`1011	B3
20	10100	B0
25	11001	B1

DATA BUCKETS	BITS CONSIDERED
B0	00
B1	01
B2	10
B3	11

DATA BUCKET	DATA RECORDS		
B0	71	8	20
B1	413	25	
B2	2	14	10
B3	55	39	11

bucket-overflow

86	**18**

Considering last two bits in the hash function

DYNAMIC HASHING

DATA RECORDS	HASH FUNCTION	BUCKET NO.
2	010	B2
14	1110	B2
55	110111	B3
71	1000111	B0
8	1000	B0
39	100111	B3
10	1010	B2
413	110011101	B1
86	1010110	B2
18	10010	B2
11	`1011	B3
20	10100	B0
25	11001	B1

DATA BUCKETS	BITS CONSIDERED
B0	000
B1	001
B2	010
B3	011
B4	100
B5	101
B6	110
B7	111

DATA BUCKET	DATA RECORDS		
B0	71	8	20
B1	413	25	
B2	2	10	18
B3	55	39	11
B6	14	86	

Considering last THREE bits in the hash function, BUCKET OVERFLOW IS RESOLVED DYNAMICALLY by introducing another bucket.

- TILL WE MEET AGAIN IN THE NEXT CLASS.....