## **UNIT - II**

## **RE-HASHING**

Re-hashing schemes use a second hashing operation when there is a collision. If there is a further collision, we *re-hash* until an empty "slot" in the table is found.

The re-hashing function can either be a new function or a re-application of the original one. As long as the functions are applied to a key in the same order, then a sought key can always be located.

If the table gets too full, then the rehashing method builds new table that is a about twice as big and scan down the entier original hashtable , computing the new hash value for each element and inserting it in the new table .

Rehashing is very expensive operation, the running time is O(N), since there are N element to rehashing and the table size roughly 2N.

Rehashing can be implemented in several ways with quadratic probing such as:

- Rehashing, as soon as the table is half full
- Rehashing only when an insertion fails.
- Rehashing when the table reaches a certain load factor.

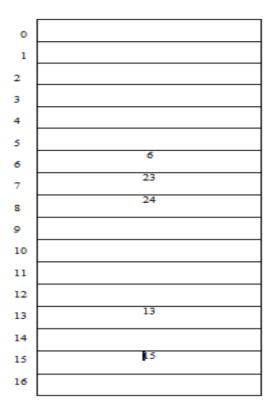
Eg: 13,15,24,6,23 are to be inserted in hash table of size 7.

0	6
1	15
2	23
3	24
4	
5	
6	13

The table will be 70% full.

A new table is created as the table is so full. The new hash function is then  $h(X) = X \mod 17$ 

III YEAR - I SEMESTER 2020-2021 INFORMATION TECHNOLOGY



## Advantages:

- Programmer does not worry about the table size.
- Simple to implement.
- Can be used in other data structures as well.