Unit-1 Advanded Data Structures

Hashing.

What is Hashing?

It is basically a method for storing and retreiving data from database in O(1) time.

Terminologies of Hashing

- *Search Keys:* a key on the basis of which data is stored and can be retreived.
- *Hash Table:* It is a data structure which stores the search keys and it provides a methodology to store data in a proper way and it is similar to an array.
- *Hash Functions:* There are diffrent methods of `Hash Functions`, some of them are:
    1. K mod 10
    2. K mod n
    3. Mid Square Method
    4. Folding Method

Example:

Search Keys: (24, 52, 91, 67, 48, 83) *Hash Function:* K mod 10 *Hash Table:*

0	
1	91
2	52
3	83
4	24
5	
6	
7	67
8	48
9	

Data is inserted in the table by using the `Hash Function` `K mod 10` where K is the `Search Key`.

Collision Resolution Techniques

What is Collision? Collision is when 2 search keys are to be placed in the same `Hash Table Column` according to the `Hash Function` used is known as `Collision`.

Example:

`Hash Function` to be used is `K mod 6` `Search Keys` are (24, 19, 32, 44)

Hash Table Created:

0	24
1	19
2	32
3	
4	
5	

Now, inserting the `Keys` in the Hash Table according to the `Hash Function` which is `K mod 6`. Therefore, `24 mod 6, remainder = 0, 19 mod 6, remiander = 1, 32 mod 6, remainder = 2, 44 mod 6, remainder = 2`

Now, there is a `collision` so there are diffrent methods to resolve collision like

- Chaining(Open Hashing)
- Open Addressing(Closed Hashing)
    1. Linear Probing
    2. Quadratic Probing
    3. Double Hashing

Chaining(Open Hashing): In Chaining we utilise some extra space by adding a `linked list` to the column which is under `collision` without fully utilising the given space.

Load Factor: It is the ratio of the number of search keys to the number of columns in the `Hash Table`.

$\alpha = {}^{\text{Number of Search Keys}}/_{\text{Number of Columns}}$

Where, α = Load Factor Number of Search Keys = Number of Search Keys Number of Columns = Number of Columns in the Hash Table

Advantages of Chaining:

- Deletion is easy to perform.
- Insertion is performed in O(1) time.(constant time)

Disadvantages of Chaining:

- Searching worst case is O(n) time.
- Extra space is required for the linked list.

Open Addressing(Closed Hashing): In this method we first utilise the space provided to us before using any extra space.

Linear Probing: In this method we use the `Hash Function` to find the `Hash Table Column` and if there is a `collision` then we move to the next column and check if it is empty or not, if it is empty then we insert the `Search Key` there otherwise we move to the next column and so on.

Formula for Linear Probing:

R(K, i) = [H(K) + i] mod n

Where, R(K, i) = Hash Function H(K) = Search Key i = Number of times we have moved to the next column also know as `prob number/collision number` n = Number of Columns in the Hash Table

Primary Clustering: It is a situation in which the `Search Keys` are inserted in the `Hash Table` in a way that they form a cluster.

Secondary Clustering: When two or more keys are competing for the same slot in the `Hash Table` then it is known as `Secondary Clustering`.

Advantages of Linear Probing:

- No Extra Space is required.

Disadvantages of Linear Probing:

- Search time is O(n) time in worst case.
- Deletion is difficult to perform.
- Primary Clustering occurs.
- Secondary Clustering occurs.

Imp Question Hashing

Question: The keys `1, 3, 12, 4, 25, 6, 18, 20, 8` are inserted into empty hash table of length 10 using open addressing with hash function `h(i) = i^2^ mod 10` and linear probing. What is the resultant hash table and find the maximum probe value?

Answer: `i = 1, 3, 12, 4, 25, 6, 18, 20, 8`

Calculating the hash values: 1^2^ mod 10 = 1 3^2^ mod 10 = 9 12^2^ mod 10 = 4 4^2^ mod 10 = 6 25^2^ mod 10 = 5 6^2^ mod 10 = 6 18^2^ mod 10 = 4 20^2^ mod 10 = 0 8^2^ mod 10 = 4

∴ `h(i) = 1, 9, 4, 6, 5, 6, 4, 0, 4`

Hash Table:

0	20
1	1
2	8
3	
4	12

5	25
6	4
7	6
8	18
9	3

`The maximum probe value is 9 for the key 8.`

Quadratic Probing: In this method we use the `Hash Function` to find the `Hash Table Column` and if there is a `collision` then we move to the next column and check if it is empty or not, if it is empty then we insert the `Search Key` there otherwise we move to the next column and so on but the difference is that we move to the next column by using the formula `i^2` where `i` is the number of times we have moved to the next column.

Formula:

R(K, i) = [H(K) + i^2^] mod n

Where, R(K, i) = Hash Function H(K) = Search Key i = Number of times we have moved to the next column also know as `prob number/collision number` n = Number of Columns in the Hash Table

Advantages of Quadratic Probing:

- No Extra Space is required.
- No Primary Clustering occurs.

Disadvantages of Quadratic Probing:

- Search time is O(n) time in worst case.
- Secondary Clustering occurs.
- No gaurantee of finding an empty slot.

Double Hashing: In this method we use the `Hash Function` to find the `Hash Table Column` and if there is a `collision` then we move to the next column and check if it is empty or not, if it is empty then we insert the `Search Key` there otherwise we move to the next column and so on but the difference is that we move to the next column by using the formula `i * Hash Function` where `i` is the number of times we have moved to the next column.

Formula:

R(K, i) = [H(K) + i * H'(K)] mod n

Where, R(K, i) = Hash Function H(K) = Search Key i = Number of times we have moved to the next column also know as `prob number/collision number` H'(K) = Second Hash Function n = Number of Columns in the Hash Table

Advantages of Double Hashing:

- No extra space is required.

- No Primary Clustering occurs.
- No Secondary Clustering occurs.

Disadvantages of Double Hashing:

- Search time is O(n) time in worst case.

Example:

H(k) = k mod 11 H'(k) = 8 - (k mod 8) [H(k) + i * H'(k)]mod 11

Keys = 20, 34, 45, 70, 56

Calculating hash values for the first hash function:

20 mod 11 = 9 34 mod 11 = 1 45 mod 11 = 1 70 mod 11 = 4 56 mod 11 = 1

Calculating hash values for the second hash function:

8 - (20 mod 8) = 8 - 4 = 4 8 - (34 mod 8) = 8 - 2 = 6 8 - (45 mod 8) = 8 - 5 = 3 8 - (70 mod 8) = 8 - 6 = 2 8 - (56 mod 8) = 8 - 0 = 8

Hash Table:

0	
1	34
2	
3	56
4	45
5	
6	70
7	
8	
9	20
10	