Hashing

Searching

• Find a location LOC of a given item from a collection of data items.

• Print some error message if item is not available.

• Three common data structures used for searching:

(1) Sorted Array [Time Complexity - O(n) for linear search]

[Time Complexity - O(logn) for binary search]

(2) Linked List [Time Complexity - O(n)]

(3) Binary Search Tree [Time Complexity - O(logn) for balanced tree]

[Time Complexity - O(n) for unbalanced tree]

• All three data structures depend on the number of data items in the file.

Hashing

- It is a searching technique that does not depend on the number of data items.
- Consider a file F having n records with a set of K search keys. F is maintained in memory using a table T and L is the set of memory addresses of T.
- In hashing, for searching an item, a hash function H is applied on the search key to get the memory address of corresponding record.
- A hash function H is a function from K to L i.e. H : K -> L.

Properties of Hash Function(H)

(1) H should be very easy to and quick to compute.

(2) Uniformly distribute the hash addresses throughout the set L.

Popular Hash Function

(1) Division Method

• Choose a number M (prime number) larger than the number n of search keys in K.

• Hash function is defined as H(K) = K mod M (if hash addresser are from 0 to M-1).

• Otherwise hash function H(K) = K mod M +1 (if hash addresser are from 1 to M).

Example:

Consider a company file consisting of 68 employee information where 4-digits employee_no (K) is used to identify each employee. Let L has 100 two-digits addresses like 01, 02,.........99. The hash function H is applied to the following employee_nos K : 3205, 7148, 2345.

Division Method

Let m=97(prime, near to 99). So,

H(3205) = (3205 mod 97) +1 = 05 H(7148) = (7148 mod 97) + 1 = 68

H(2345) = (2345 mod 97) + 1 = 18 **Hash Addresses are 05, 68, 18.**

(2) Midsquare Method

• The key is squared.

• The hash function can be defined as H(K) = Q where Q is obtained by deleting digits from both ends of K^2.

Example:

Consider a company file consisting of 68 employee information where 4-digits employee_no (K) is used to identify each employee. Let L has 100 two-digits addresses like 00, 01,.........99. The hash function H is applied to the following employee_nos K : 3205, 7148, 2345.

<u>Midsquare Method</u>

K:	3205	7148	2345
K^2:	10272025	51093904	5499025
H(K):	72	93	99 [Considering 4^{th} & 5^{th} digits from right]

Hash Addresses are 72, 93, 99.

(3) Folding Method

- Partition the search key K into a number of parts K_1, K_2,....K_r where each part except possibly the last, has the same number of digits as the required address.
- Then add all parts together, ignoring the final carry.
- The hash function H is defined as $H(K)=K_1+K_2+......+K_r$.

Example:

Consider a company file consisting of 68 employee information where 4-digits employee_no (K) is used to identify each employee. Let L has 100 two-digits addresses like 00, 01,.........99. The hash function H is applied to the following employee_nos K : 3205, 7148, 2345.

Folding Method

Partition K into 2 parts and add the parts ignoring final carry.

H(3205) = 32 + 05 = 37 H(7148) = 71 + 48 = 119 = 19

Ignoring 1

H(2345) = 23 + 45 = 68

Hash Addresses are 37, 19, 68.

Collision

- It is possible to have the same hash address for two different keys K1 and K2. This situation is called collision.
- Suppose a new record with search key K is to added to a file F but the hash address H(K) is already occupied. So collision is occurred.
- To resolve collision, two techniques are given below:

  (1) Open Addressing (linear Probing)

  (2) Chaining

Open Addressing (Linear Probing)

• Suppose a new record R with search key K is to be added to the memory table T. But the memory location with address H(K) = h (i.e.T[h]) is already occupied. So Collision is happened.

• In open addressing, the collision can be resolved by assigning R to the first available location following T[h] i.e. T[h+1],T[h+2],................[Linearly Search T].

Example:

Record:	A	B	C	D	E	X	Y	Z
H(K):	4	8	2	11	4	11	5	1

Address:	1	2	3	4	5	6	7	8	9	10	11
Table T:	X	C	Z	A	E	Y	--	B	--	--	D

Chaining

This technique resolves collision by maintaining two separate tables in memory.

(1) Table T maintains two separate fields for each data items.

(a) Info: for record/information

(b) Link: link of all records in T with same hash address.

(2) Table List contains the pointers to the linked list in T.

Example:

Record:	A	B	C	D	E	X	Y	Z
H(K):	4	8	2	11	4	11	5	1

Result is given on the next slide.

LIST

	LIST
1	8
2	3
3	
4	5
5	7
6	
7	
8	2
9	
10	
11	6

TABLE T

INFO	LINK	
A	0	1
B	0	2
C	0	3
D	0	4
E	1	5
X	4	6
Y	0	7
Z	0	8
		9
		10
		11

End!!!