6.4 HASH TABLES

There are other types of tables which help us to retrieve information very efficiently. The ideal *hash table* is merely an array of some constant size; the size depends on the application where it will be used. The hash table contains key values with pointers to the corresponding records. The basic idea of a hash table is that we have to place a key value into a location in the hash table; the location will be calculated from the key value itself. This one-to-one correspondence between a key value and an index in the hash table is known as *address calculation indexing* or more commonly *hashing*. In the present section, we will discuss hashing techniques and their related issues.

6.4.1 Hashing Techniques

The main idea behind any hashing technique is to find a one-to-one correspondence between a key value and an index in the hash table where the key value can be placed. Mathematically, this can be expressed as shown in Figure 6.6, where K denotes a set of key values, I denotes a range of indices and H denotes the mapping function from K to I.

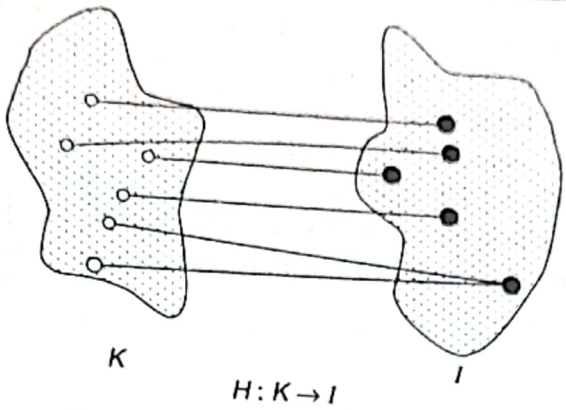

Figure 6.6 Concept of hashing.

It may be noted that the mapping is subjective, that is all key values are mapped into some indices and more than one key value may be mapped into an index value. The function that governs this mapping is called the *hash function*. A particular hashing technique uses a particular hash function. The hash function plays a dominant role in hashing techniques. There are two principal criteria in deciding a hash function $H:K \rightarrow I$ as follows:

1. The function H should be very easy and quick to compute.
2. The function H should as far as possible give two different indices for two different key values.

As an example, let us consider a hash table of size 10 whose indices are 0, 1, 2, ..., 8, 9. Suppose a set of key values are: 10, 19, 35, 43, 62, 59, 31, 49, 77, 33. Let us assume the hash function H is as stated below:

- Add the two digits in the key.
- Take the digit at the unit place of the result as the index; ignore the digit at the tenth place, if any.

Using this hash function, the mappings from key values to indices and to hash table are shown in Figure 6.7. In this example, for the given set of key values, the hash function does

K	I
10	1
19	0
35	8
43	7
62	8
59	4
31	4
49	3
77	4
33	6

$H: K \rightarrow I$

0	19
1	10
2	
3	49
4	59, 31, 77
5	
6	33
7	43
8	35, 62
9	

Hash table

Figure 6.7 Example of hashing.

not distribute them uniformly over the hash table; some entries are there which are emp in some entries more than one key value needs to be stored. Allotment of more than value in one location in the hash table is called *collision*. We have found three collis 62, 31 and 77 in the above-mentioned example.

It can be noted that $| K | = | I |$, that is, the number of key values is the same as of the hash table, but this is not the case always. In general, $| K | > | I |$.

The following are some hash functions which are very common and popularly ap various applications.

Division method

One of the fast hashing functions, and perhaps the most widely accepted, is the division which is defined as follows:

Choose a number h larger than the number N of keys in K. The hash function H defined by

$$H(k) = k(\text{MOD } h) \qquad \text{if indices start from 0}$$

or

$$H(k) = k(\text{MOD } h) + 1 \qquad \text{if indices start from 1}$$

where $k \in K$, a key value. The operator MOD defines the modulo arithmetic operation. is equal to the remainder of dividing k by h. For example, if $k = 31$ and $h = 13$ then

$$H(31) = 31(\text{MOD } 13) = 5$$

or

$$H(31) = 31(\text{MOD } 13) + 1 = 6$$

The number h is usually chosen to be a prime number or a number without small divisor this usually minimizes the number of collisions. Generally, h is a prime number and e the size of the hash table.

Midsquare method

Another hash function which has been widely used in many applications is the mi method. The method is defined as follows:

The hash function H is defined by $H(k) = x$, where x is obtained by selecting an appr number of bits or digits from the middle of the square of the key value k. This selection depends on the size of the hash table. It needs to be emphasized that the same criteria be used for selecting the bits or digits for all of the keys.

As an example, suppose the key values are of the integer type, and we require addresses. Our selection criteria are to sel 3 digits at ions starting from the most digit in the square. Let us see t calcul distinct keys and w hash function, as defined above:

k	:	1234
k^2	:	1522756
$H(k)$	:	525