Collision Free Hashing

Gopi Manoj Vuyyuru[1], Atcha Naga Mounika[2], Teja Veerendra Marupudi[3]

[1]Tata Consultancy Services, Bangalore, Karnataka, India
[2]Vignan's Lara Institute of Technology & Science, Guntur, Andhra Pradesh, India
[3]Tata Consultancy Services, Hyderabad, Telangana, India

---***---

Abstract- Out of distinct problems in the domain of computer sciences, hashing has its own importance. Hashing means mapping of data into a hash table with the help of hash function. It is widely adopted in the fields of database management and cryptography. There are numerous hash functions designed in order to map the data into a hash table. While mapping the data into a hash table, there is a chance that two data elements have the same key in this case collisions occur. In this paper we have proposed a new algorithm which eliminates collisions if we don't have any duplicate data.

Keywords– Hashing, Hash Function, Collisions, Hash Table, Division Method, Multiplication Method

1. INTRODUCTION

Hashing is the process of computing hash value with the help of a mathematical function called a hash function. The hash function is any mathematical function that takes some finite numerical value and generates a hash value. The hash value or simply hash is used as an index to store the original numerical data into a data structure named as hash table. Hash table is a data structure which stores the data in an associative manner where each value has its own individual index. [1]

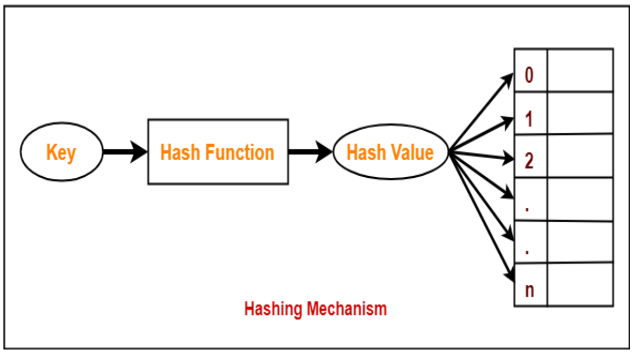


Figure 1: Hashing Mechanism

Hashing provides the fastest way of data retrieval than any other data structure. It is also faster than retrieving data from arrays and lists. Due to its speed, it is widely used in databases to retrieve data. It is also extensively used in many encryption algorithms to encrypt the data. But the major drawback is that hashing cannot be used for sorting of data. Sorting means arranging data in a particular manner whereas hashing arranges the data based on the hash function. That's the prime reason why there is no relationship among the data. [2]

In this paper, a new hash function is proposed to eliminate the collisions while mapping the data into a hash table. Initially, the paper discusses few hash functions briefly. Then in upcoming sections, it is clearly shown how this hash function is implemented and finally, we compare the proposed method with existing hash functions.

2. OVERVIEW OF SOME WELL KNOWN HASH FUNCTIONS

Let's look at some available hash functions to map the data into hash table.

2.1 Division Method

The division method is one of the easiest and simple hash functions, in which we perform modulo of element (E) with size of hash table (N) to obtain the key. So, the hash function is as follows

Key = E mod N

Let N=7, and the elements are as follows: 26,3,11,8,47

K(26) = 26 mod 7 = 5

K(3) = 3 mod 7 = 3

K(11) = 11 mod 7 = 4

K(8) = 8 mod 7 = 1

K(47) = 47 mod 7 = 5

After mapping the data, the hash table is as follows:

Index	Element
0	
1	8
2	
3	3
4	11
5	26, 47
6	

Figure 2: Hashing table after performing division method.

From the above it is clear that at index 5, there is collision. So, the best choice of N is the nearest prime number greater than N.

2.2 Multiplication Method

In multiplication method the hash function is as follows:

H(E) = |_ M[EA mod 1] _|

EA mod 1 = EA - |_EA_|

Where,

E = element,

M = nearest prime number greater than size of array

A = 1/ Ø

Ø is golden ratio and the value is as follows

golden ratio Ø = (1 + sqrt (5)) / 2 = 1.618033989

So, A = 1/ 1.618033989 = 0.618033989.

Let M=7, and the elements are as follows : 26,3,11,8,47

H(26) = |_7[26 * 0.618033989 mod 1]_| = 0

K(3) = |_7[3 * 0.618033989 mod 1]_| = 5

K(11) = |_7[11 * 0.618033989 mod 1]_| = 5

K(8) = |_7[8 * 0.618033989 mod 1]_| = 6

K(47) = |_7[47 * 0.618033989 mod 1]_| = 0

After mapping the data, the hash table is as follows:

Index	Element
0	26, 47
1	
2	
3	
4	
5	3, 11
6	8

Figure 3: Hashing table after performing multiplication method.

From the above it is clear that at index 0, 5, there is a collision.

3. EXPLANATION OF COLLISION FREE HASHING

The major idea of this algorithm revolves around the size of the greatest element in the data. If we know the size of the greatest element then only, we can implement this algorithm. Now we apply the hash function for the element and map the element into the hash table. The implementation of this algorithm is clearly explained in the upcoming steps.

3.1 Working of Collision free hashing

Let n be the number of elements given, E be the element and i be the size of the largest element. Then apply the following hash function to map each element into the hash table. The size of hash table is equal to N.

N is given as N = $(10^{i-1} * 9)$

the values of N are as follows: 9, 90, 900 …

The hash function is as follows:

Key (E) = E Mod N

Where,

E = element to perform the hashing

N = size of hash table

After performing the above hash function, we will obtain key (K) which is used to map the element into hash table.

Let number of elements be n=7, and elements be 90,8,101, 750, 62,650 and size of maximum element (i) =3.

The size of hash table (N) = $(10^{i-1} * 9) = (10^{3-1} * 9)$ = 900,

Key(90) = 90 Mod 900 = 90

Key(8) = 8 Mod 900 = 8

Key(101) = 101 Mod 900 = 101

Key(750) = 750 Mod 900 = 750

Key(62) = 62 Mod 900 = 62

Key(650) = 650 Mod 900 = 650

Index	Element
0	
….	
8	8
….	
62	62
….	
90	90
….	
101	101
….	
650	650
….	
750	750
….	
900	

Figure 4: Hashing table after performing Collision free method.

Note: The following are two cases where collisions might happen.

i) When we have element equal to that of size of hash table then we have collision.

Key(N) = N Mod N = 0;

Key(0) = 0 Mod N = 0;

ii) The other case where collision happens is duplicate elements. If we have duplicate elements in the data then collision occurs.

But the above two case are commonly observed in other hashing methods.

3.2 Algorithm

The algorithm consists of 3 parts:

i) Reading of data and size of maximum element.

1. Enter number of elements: n

2. Read elements

3. Enter size of maximum element (i)

ii) Determining the size of hash table (N)

The size of hash table N = (10^{i-1} * 9)

iii) Apply the hash function for the elements to get key

Key (E) = E Mod N

After obtaining key from above hash function we store the element into the hash table.

4. COMPARISION OF COLLISION FREE HASHING WITH OTHER HASHING TECHNIQUES

Now let's compare the efficiency of proposed algorithm with other hashing techniques. For this let us consider the following example.

Let n = 20 and elements are as follows:

730,59,6,176,95,70,11,79,221,111,410,570,699,372,692,19,1000,672,512,890.

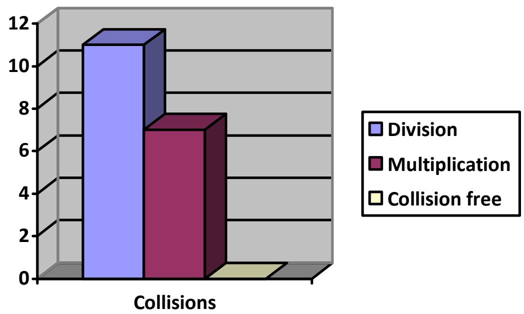
After performing the hashing with other techniques, the keys for the elements are as follows:

Element	Division method	Multiplication Method	Collision free hashing
730	10	3	730
59	19	10	59
6	6	16	6
176	16	17	176
95	15	16	95
70	10	6	70
11	11	18	11
79	19	18	79
221	1	13	221
111	11	13	111
410	10	9	410

570	10	6	570
699	19	0	699
372	12	20	372
692	12	15	692
19	19	17	19
1000	0	0	1000
672	12	7	672
512	12	9	512
890	10	1	890

Table 1: Comparison between collision free hashing and other hashing techniques.

From the above table we can see that, using division method there are 11 collisions and then using multiplication method we have 7 collisions. Where as in the proposed method we don't have any collisions.



Graph 1: Graphical comparison of collisions between free hashing and other hashing techniques

The major advantage of using this technique is searching of element will take exactly O(1) time where as other techniques take some extra time in case of collisions. Also there is one more advantage is that insertion of new data also take exactly O(1) time. In this method we can see that as you go deep the hash table the value of elements increases. But the major disadvantage is this algorithm requires more space than the other techniques.

5. CONCLUSION

In this paper, a new hashing method is presented which completely diminishes the collisions that usually occur in other hashing techniques. An illustration is presented to show how this algorithm works and also a comparison between various techniques is made. It is shown that using this approach we can achieve O(1) in searching and also inserting new elements into the hash table. Also, in this method, we have established a relationship between the elements in the hash table. In the future scope, this algorithm is used in the fields of databases and also cryptography.

REFERENCES

[1] wikipedia.org. 2020. Hash Function. [online] Available: <https://en.wikipedia.org/wiki/Hash_function> [Accessed 07 April 2020].
[2] M. Singh and D. Garg, "Choosing Best Hashing Strategies and Hash Functions," 2009 IEEE International Advance Computing Conference, Patiala, 2009, pp. 50-55.

[3] "Hashing Functions", Csee.umbc.edu, 2020. [Online]. Available:https://www.csee.umbc.edu/courses/undergraduate/341/fall99/frey/Lectures/Hashing/hashFunc.html. [Accessed: 10- Apr- 2020]
[4] Cseweb.ucsd.edu, 2020. [Online]. Available: https://cseweb.ucsd.edu/~mihir/papers/inchash.pdf. [Accessed: 10- Apr-2020]
[5] Data structures using c and c++ by Yedidyah langsam,Aaron M. Tenenbaum second Indian printing (Prentice Hall of India private limited), New Delhi-110001
[6] Data structure using C by Reema Thareja(first published 2011 by Oxford University Press) in INDIA Ram Book Binding House,New Delhi.