Hashing in Data Structures

Mayank Pratap Singh

July 31, 2024

1 Introduction

Hashing is a technique used to uniquely identify a specific object from a group of similar objects. It is used to store and retrieve data efficiently using a hash function, which converts a given key into a specific address in the hash table.

2 Hash Function

A hash function $h(k)$ is used to compute the hash value of a key k. The value of $h(k)$ is the index at which the data associated with the key k is stored in the hash table.

2.1 Division Method

One of the most common hash functions is the division method:

$$h(k) = k \mod m$$

where m is the size of the hash table.

3 Example of Hash Function

Consider a hash table of size 10 and the keys: 23, 74, 56, 95, 47, 69. Using the division method, the hash values are:

$$\begin{aligned}
h(23) &= 23 \mod 10 = 3 \\
h(74) &= 74 \mod 10 = 4 \\
h(56) &= 56 \mod 10 = 6 \\
h(95) &= 95 \mod 10 = 5 \\
h(47) &= 47 \mod 10 = 7 \\
h(69) &= 69 \mod 10 = 9
\end{aligned}$$

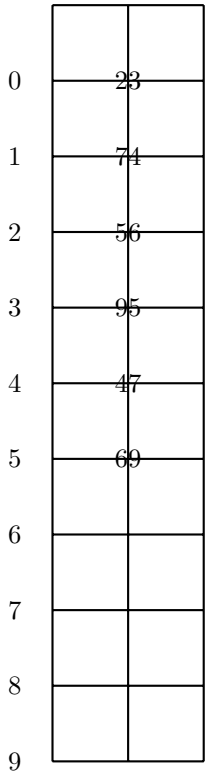

Figure 1: Hash Table with keys hashed using division method

4 Handling Collisions

When two keys hash to the same index, it is called a collision. There are several techniques to handle collisions:

4.1 Open Addressing

In open addressing, all elements are stored in the hash table itself. When a collision occurs, a probing sequence is used to find the next empty slot. The common probing techniques are:

1. Linear Probing: In linear probing, if a collision occurs at index i, we check the next slot $i+1$, $i+2$, and so on until an empty slot is found.
2. Quadratic Probing: In quadratic probing, if a collision occurs at index i, we check the slots at $i+1^2$, $i+2^2$, $i+3^2$, and so on.
3. Double Hashing: In double hashing, we use a second hash function to calculate the interval between probes.

4.2 Chaining

In chaining, each slot in the hash table points to a linked list of records that have the same hash value.

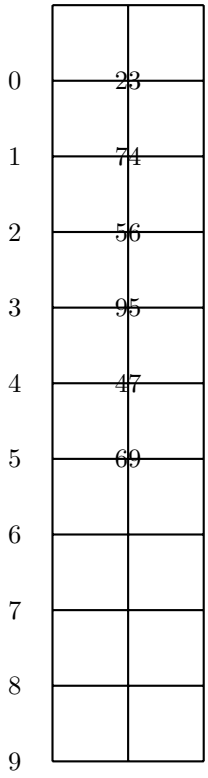

Figure 2: Chaining to handle collisions in a hash table

5 Load Factor

The load factor λ is defined as the ratio of the number of elements in the hash table to the size of the table:

$$\lambda = \frac{n}{m}$$

A high load factor can lead to more collisions and decreased performance.

6 Example Problem

Write a function to count the frequency of each character in a string using a dictionary (hash table in Python).

```
def count_characters(string):
    char_count = {}
    for char in string:
        if char in char_count:
            char_count[char] += 1
        else:
            char_count[char] = 1
    return char_count

string = "Mayank Pratap"
result = count_characters(string)
print(result)
```

7 Complexity Analysis

7.1 Time Complexity

The time complexity for inserting, deleting, and searching in a hash table is $O(1)$ on average. However, in the worst case, it can be $O(n)$ if all elements hash to the same index.

7.2 Space Complexity

The space complexity is $O(n)$ where n is the number of elements in the hash table.