DSA Assignment1 Skip List

CS21B059 Chandradithya J CS21B025 K E Nanda Kishore

April 2023

1 Introduction

A skip list is a probabilistic data structure that allows for efficient search, insertion, and deletion of elements in a sorted list. Its average time complexity is determined through a probabilistic analysis.

2 Functionality

Searching through a linked list takes O(n) time. The same time is taken even for searching through a sorted linked list as we cannot jump from one node to another, if they are not linked.

For faster searching, skip lists are used. They implement the functionality of being able to skip across elements, which reduces search time. This is done by having multiple levels of linked lists in a single skip list.

Each level has a lesser number of elements than the one below it. The bottom level is a regular linked list, while the others are skipping links that contain certain elements, and skip over the rest. These allow for fast navigation between elements that are far away in the bottom level. The idea behind this is to allow for quick traversal to the desired element, reducing the average number of steps needed to reach it.

3 Implementation

Skip lists are implemented using a technique called "coin flipping". In this technique, a random number is generated for each insertion to determine the number of layers the new element will occupy. This means that, on average, each element will be in log(n) layers, where n is the number of elements in the bottom layer.

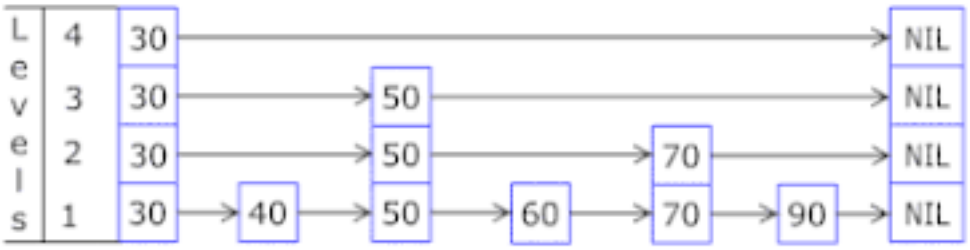

Figure 1: Skip List

While inserting an element, a coin is flipped, as mentioned above, and the number of levels it will occupy is determined. It is placed at the appropriate position in each level, and if a level isn't present, a new one will be created.

While deleting an element, the instance of that element in each level is deleted.

While searching, we start at the topmost level, and if we encounter a number greater than the one required, we go down to the level below, and continue our search there.

Skip lists have an average time complexity of O(log n) for search, insertion and deletion, which is similar to that of balanced trees, such as AVL trees and red-black trees, but with the advantage of simpler implementation and lower overhead.

4 Advantages of a Skip List

- Easy to implement compared to the hash table and the binary search tree.
- Quick insertion of a new node.
- As the number of elements increases, the possibility of the worst case occurring decreases.
- Requires only θ(log n) time in the average case for all operations.
- Quick and straightforward searching.

5 Disadvantages of a Skip List

- Requires a greater amount of memory than a balanced tree.
- Reverse search is not permitted.
- Skip lists are not cache-friendly because they don't optimize the locality of reference

6 Height of a Skip List

Earlier, we said that there will be O(log n) levels in a skip list. How can we know this if each level is created probabilistically?

The probability P_i that an element in a skip list with n total elements gets up to level i is

$$P_i = \frac{1}{2^i} \tag{1}$$

If the probability is $\frac{1}{2}$, then to create a new level, we're tossing a coin for each element to see if it should be included. So, the probability that at least one of the n elements gets to level i is

$$P_i \leq \frac{1}{2^i} \tag{2}$$

P_i equals 1, when i equals log(n).

Hence, the height of a skip list is O(log n).

7 Space Complexity Analysis

Number of nodes in i^{th} level is

$$\frac{n}{2^i} \tag{3}$$

So, the total number of nodes in the skip list (where h is the height of the skip list) is

$$\sum_{i=0}^{h} \frac{n}{2^i} = 2n \tag{4}$$

As this may probabilistically go infinitely.

Hence, the space complexity is O(n).

8 Time Complexity Analysis

In this section, we shall compute the time complexity for a search operation on a skip list. This is done by calculating the number of steps in a search operation.

The expected number of nodes at the j^{th} level of a skip list is 1 plus the probability that we've reached on the previous level plus probability in this level.

$$C(j) = 1 + \frac{1}{2}C(j-1) + \frac{1}{2}C(j) \tag{5}$$

$$C(j) = 2 + C(j-1) = 2 + 2 + C(j-2) \tag{6}$$

There are log n levels.

$$C(j) = 2log(n) \tag{7}$$

Hence, the time complexity of a search operation on a skip list is O(log n).

9 Pseudo Codes

Below is the pseudo code for the search operation:

```
Search(key):
    p = top-left node in S
    while (p.below != null) do          //Scan down
        p = p.below
        while (key >= p.next) do        //Scan forward
            p = p.next
    return p
```

Below is the pseudo code for the insert operation:

```
Insert(key):
    p = Search(key)
    q = null
    i = 1
    repeat
        i = i + 1                       //Height of tower for new element
        if i >= h
            h = h + 1
            createNewLevel()            //Creates new linked list level
        while (p.above == null)
            p = p.prev                  //Scan backwards until you can go up
        p = p.above
        q = insertAfter(key, p)         //Insert our key after position p
    until CoinFlip() == 'Tails'
    n = n + 1
    return q
```

Below is the pseudo code for the delete operation:

```
Delete(key):
    Search for all positions p_0, ..., p_i where key exists
    if none are found
        return
    Delete all positions p_0, ..., p_i
    Remove all empty layers of skip list
```