Homework 1 – Skip lists
ENSEA/FAME Computer Science

Due: Tuesday, April 21th 2020

Introduction

A skip list is a probabilistic data structure, invented in 1989 by William Pugh. It is used to represent an ordered sequence. It can be thought as many layers of linked lists. The bottom layer is an ordinary ordered linked list (see figure 1). Each higher layer acts as an express lane.

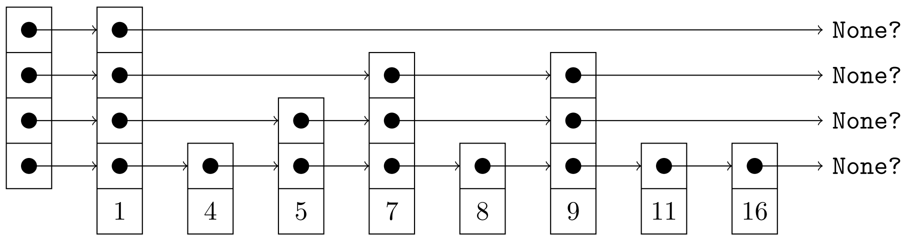

Figure 1: `skippy` the skip list

More precisely, a skip list is a collection of nodes. Each node contains a list of pointers, and all but the first one have a label (in case of figure 1 an integer, but the labels can be of any ordered type). The first node is called the *head*. The *height* of the skip list is the maximal number of pointers in a node. Here is how we implement nodes.

```python
class SkipListNode:
    def __init__(self, label, height):
        self.label = label
        self.next = [None] * height
```

For example, the skip list in figure 1 contains 9 nodes. Here is our definition of a skip list.

```python
class SkipList:
    def __init__(self):
        self.head = SkipListNode(None, 0)
```

When we instantiate a skip list, its head carries no information (it is a node of height 0 and its label is `None`).

List operations

Suppose we want to know if 13 belongs to our skip list skippy. We start in the topmost level of the head (see figure 2). We go through the list in this level until we find node with the largest element that is smaller than 13 (or until we reach `None`). We then go to the level below and search again for node with the largest element that is smaller than 13, but this time we begin the search from the node we found in the level before.

We repeat this process until we reach the bottom level. The node found in the bottom level will be the largest element that is smaller than 13 in the whole list.

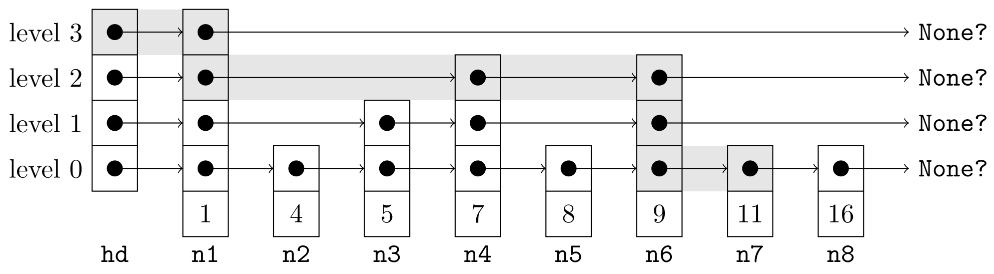

Figure 2: Finding a path towards 13 in `skippy`

Question 1. Write a method `def findpath(self, label)` in the class `SkipList` which returns a list of nodes in each level that contains the greatest value that is smaller than `label`.

For example `skippy.findpath(13)` returns `[n7, n6, n6, n1]`. Also `skippy.findpath(7)` returns `[n4, n4, n4, n1]` whereas `skippy.findpath(0)` returns `[hd, hd, hd, hd]`.

Question 2. Write a method `def mem(self, x)` in the class `SkipList` which returns `True` is `x` is in the skip list `self`, and `False` otherwise.

Question 3. Figure 3 shows the effect of deleting an element in a skip list. In the class `SkipList`, write a method `def delete(self, x)` which deletes `x` in the skip list `self`. In case `x` does not belong to `self`, this method has no effects on `self`.

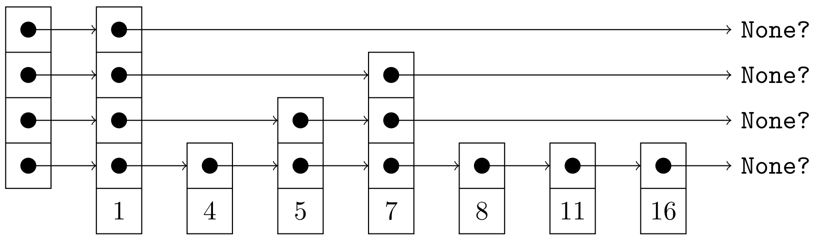

Figure 3: Deleting 9 from `skippy`

To insert a node with value x in a skip list, we proceed roughly in the following way:

- we choose randomly an integer $\ell \geqslant 1$, according to a geometric law of parameter 1/2, that is $P(\ell = 1) = 1/2$, $P(\ell = 2) = 1/4$, and more generally $P(\ell = n) = 1/2^n$.
- we insert a `SkipListNode` of label `x` and height ℓ
- we connect this node with the other in the lists. In case where ℓ is greater than the height of the skip list, we have to add pointers in the head node (see figure 4 for an example).

Question 4. Write the method `def insert(self, x)` in the class `SkipList` which inserts a node of label `x` in `self`.

Question 5. Create a skip list containing the elements 1, 4, 5, 7, 8, 9, 11 and 16. Give the height of this skip list.

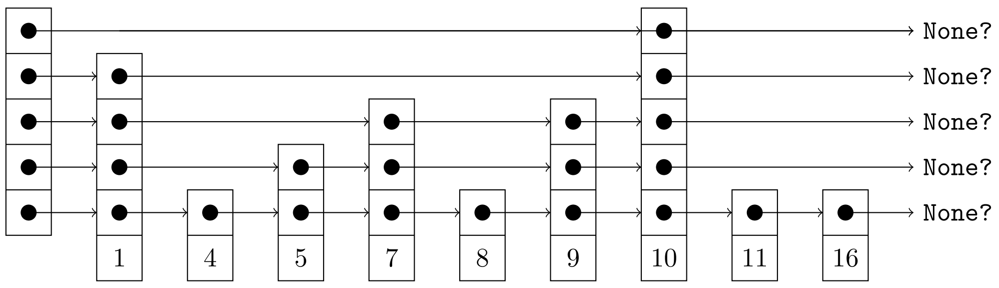

Figure 4: Inserting 10 to `skippy`

Complexity

Suppose we have a skip list with $n+1$ nodes. We call X_i the height of each node. Recall that X_0 is the height of the head, there, it is the height of the whole list. Moreover, $X_1, \dots, X_n$ are chosen independently according to the geometric law of parameter 1/2.

$$\forall i \in \{1, \dots, n\}, \forall k \geqslant 1, \quad P(X_i = k) = \frac{1}{2^k}.$$

Question 6. Explain why $X_0 = \max(X_1, \dots, X_n)$.

Question 7. Let h be a positive integer. Show that

$$P(X_0 \leqslant h) = \left(1 - \frac{1}{2^h}\right)^n.$$

Question 8. Deduce that

$$P(X_0 \geqslant C \log_2 n) \underset{n\to\infty}{\sim} \frac{1}{n^{c-1}}$$

Conclude that the height of a skip list is $O(\log_2 n)$ with high probability.

Question 9. Show that the complexity of the method `findpath` is $O(\log_2 n)$ with high probability. Deduce from this result the complexities of insertion, deletion, and searching.