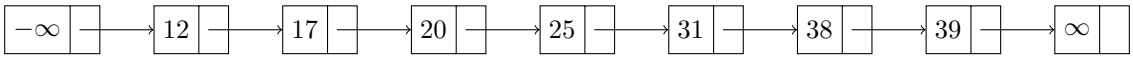
CSCI 104L Lecture 19: Skip Lists

Linked Lists

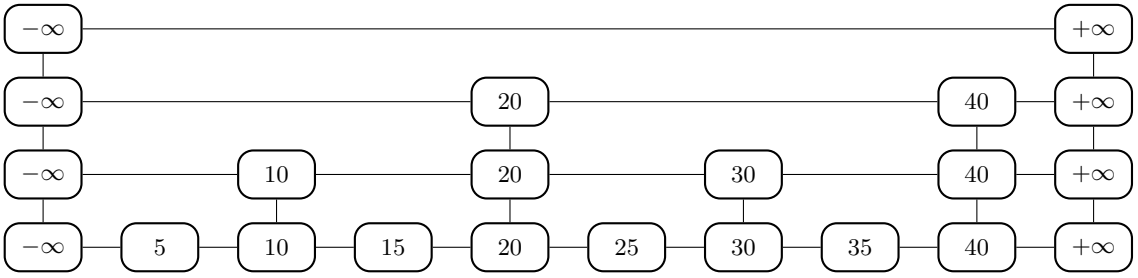
Suppose we are given a linked list of distinct integers. We know that $-\infty$ and $+\infty$ are in the list and the list is sorted least to greatest. We want to write find(k), which returns the largest key whose value is *at most* k.



If we need to find(k) more quickly, we could add extra links to the linked list that allow us to skip past nodes, such as links that skip the next node, and links that skip the next three nodes.

Skip Lists

The following data structure is known as a *Skip List*



Each layer is sorted, and contains every even element of the layer beneath it, and includes $-\infty$ and $+\infty$. Each node in the layer has a pointer to the node beneath it. The top layer has only $-\infty$ and $+\infty$.

Question 1. How would you perform a find on a linked list with multiple layers, as shown above?

Question 2. What's the runtime of your find operation?

Question 3. Suppose I want to add a key 38 to the skip list above. How do we update the structure to do this? In particular, how do we decide how high up to make 38? Do we adjust other columns? Why or why not?

The above example is a *perfect skip list*, but as you can see, this is non-ideal for inserting keys.

The solution is to add *randomness* to the construction of the skip list. Each list will contain approximately half of the elements of the list below it. If we do this, it is called a *skip list*.

Operations

Question 4. How would you remove from a skip list?

To insert, add the node at the bottom level, and then flip a coin to determine if this node will also show up on the next level. As long as the answer is yes, you flip the coin again to determine if it shows up on the level after that, until you get a "no" answer.

Question 5. How many levels will a specific node appear in, in expectation?

Question 6. What is the probability a specific node will appear in level $\log n$?

Question 7. In expectation, how many nodes will appear in level $\log n$?

Question 8. What is the probability a specific node will appear in level $c \log n$, for some constant c?

For this reason, we say that the maximum level of a skip list is $\Theta(logn)$ with high probability. Some implementations of skip lists therefore set a maximum level of $\log n$. You can also tweak p, the probability that a node exists on the next level: common choices are $\frac{1}{2}$ and $\frac{1}{4}$.

Runtime

Question 9. How long does find(k) take in the worst case?

To figure out how long find(k) takes in expectation, work backwards by starting at k and tracing the path it took, answering the following questions.

Question 10. What is the probability that k exists on a higher level?

If it doesn't exist on a higher level, follow the prev pointer to the prior node, and keep tracing backwards until you find a node that also has an up pointer.

Question 11. How many nodes do we visit on the bottom level, in expectation?

Question 12. In expectation, how many nodes do we visit overall?

Every operation is dominated by the time it takes to find the node in question. Thus, everything has worst-case $\Theta(n)$, but average-case $\Theta(\log n)$. We cannot do an amortized analysis (or rather, the amortized will be the same as the worst-case).

Question 13. How many nodes does a Skip List with n keys have, in expectation?

Most of the implementation is straight-forward, as you'll maintain next, prev, up, and down pointers for everything. Insert is the most annoying function to implement.

Question 14. What kind of problems will arise when inserting into a skip list?