# Design and Analysis of Algorithms: Lecture 7

Ben Chaplin

## Contents



# 1 Skip Lists

## 1.1 Intuition

We can understand skip lists as offering the same benefits of an express train. Consider a subway with stops:

$$14 \leftrightarrow 23 \leftrightarrow 34 \leftrightarrow 42 \leftrightarrow 59 \leftrightarrow 66 \leftrightarrow 72 \leftrightarrow 79$$

We can imagine two trains, one stopping at each station, and the other skipping stops:

$$\begin{array}{llllllll} 14 \leftrightarrow & & 34 \leftrightarrow & & 59 \leftrightarrow & & 72 & \\ 14 \leftrightarrow & 23 \leftrightarrow & 34 \leftrightarrow & 42 \leftrightarrow & 59 \leftrightarrow & 66 \leftrightarrow & 72 \leftrightarrow & 79 \end{array}$$

Then, if we start at station 14 and want to get to station 66, we can save some stops by taking the "express" $14 \to 34 \to 59$, switch down to the "local", then finish $59 \to 66$.

An even better system might have an "double-express" train:

$$\begin{array}{llllllll} 14 \leftrightarrow & & & & 59 & & & \\ 14 \leftrightarrow & & 34 \leftrightarrow & & 59 \leftrightarrow & & 72 & \\ 14 \leftrightarrow & 23 \leftrightarrow & 34 \leftrightarrow & 42 \leftrightarrow & 59 \leftrightarrow & 66 \leftrightarrow & 72 \leftrightarrow & 79 \end{array}$$

...and so on.

## 1.2 Data structure

Skip lists maintain a dynamic set of $n$ elements with $O(\log n)$ time per operation expectation, **with high probability**.

**Definition.** A parametrized event $E_\alpha$ occurs **with high probability** if, for any $\alpha \geq 1$, there is an appropriate choice of constants for which $E_\alpha$ occurs with probability at least $1 - \dfrac{c_\alpha}{n^\alpha}$.

It keeps all elements in a linked list at "layer 0" or $L_0$. It maintains layers of linked lists above, each of which may or may not contain one of the elements below. If an element is in $L_i$, then it is also in $L_{i-1}, L_{i-2}, \ldots, L_0$. Each node in one of the linked lists stores:

- a value
- a pointer to the next element of the list in the same layer, if not at the end
- a pointer to the previous element of the list in the same layer, if not at the beginning
- a pointer to the same element of the list in the above layer, if it exists
- a pointer to the same element of the list in the below layer, if not at $L_0$

### 1.3 Algorithms

**Algorithm 1** SEARCH

**Input**
$S, x$ the skip list, and value for which to search

```
e = S.head
while true do
    while e.next ≠ null and e.next.value < x do
        e = e.next
    end while
    if e.value = x then
        return true
    end if
    if e.down ≠ null then
        e = e.down
    else
        return false
    end if
end while
```

**Algorithm 2** INSERT

**Input**
$S, x$ the skip list, and value to insert

```
e = SEARCH-TRAVERSE(S, x)    ▷ same as SEARCH, but stop before returning false and just return e
f = new Node(x, prev : e.prev, next : e.next)
e.next.prev = f
e.next = f
r = FLIP-COIN()
while r = heads do
    e.up = new Node(x, prev, next)
    e = e.up
    r = FLIP-COIN()
end while
```

**Algorithm 3** DELETE

**Input**
$S, x$ the skip list, and value to delete

```
e = SEARCH-TRAVERSE(S, x)    ▷ same as SEARCH, but stop before returning false and just return e
e.prev ? e.prev.next = e.next
e.next ? e.next.prev = e.prev
while e.up ≠ null do
```

```
    e = e.up
    e.prev ? e.prev.next = e.next
    e.next ? e.next.prev = e.prev
end while
```

## 1.4 Analysis

**Lemma 1.** *The number of layers after inserting $n$ elements into a skip list is $O(\log n)$, with high probability.*

*Proof.*

$$\begin{aligned}
Pr(\text{number of layers } \neq c \cdot \log n) &= Pr(\text{number of layers } > c \cdot \log n \; \forall c \in \mathbb{R}^+) \\
&= Pr(\text{some insert flipped heads } > c \cdot \log n \text{ times}) \\
&\leq n \cdot Pr(\text{a particular insert flipped heads } > c \cdot \log n \text{ times}) \\
&= n \left(\frac{1}{2}\right)^{c \log n} \\
&= n \cdot \frac{1}{n^c} \\
&= \frac{1}{n^{c-1}}
\end{aligned}$$

Therefore, the probability that the number of layers in the skip list after $n$ inserts is $O(\log n)$ is $1 - \frac{1}{n^{c-1}}$. □

**Theorem 1.** *Searching for an element (**Algorithm 1**) in an $n$-element skip list costs $O(\log n)$ time, with high probability.*

*Proof.* Consider a search traversal backwards (from $L_0$ up/left). For each node:

- If there is a node "up", we flipped heads
- If there is no node "up", we flipped tails

$$\begin{aligned}
\text{The number of "up" moves } &< \text{ the number of levels} \\
&\leq c \log n \quad \text{w.h.p.} \qquad\qquad \text{by } \textbf{Lemma 1}
\end{aligned}$$

Therefore, w.h.p., the number of moves is at most the number of coin flips until we get $c \log n$ heads. Let $a$ be arbitrarily large. Say we flip the coin $a \cdot c \log n$ times.

$$\begin{aligned}
Pr(\text{exactly } c \log n \text{ heads}) &= \binom{a \cdot c \log n}{c \log n} \left(\frac{1}{2}\right)^{c \log n} \left(\frac{1}{2}\right)^{(a-1) c \log n} \\
Pr(\text{at most } c \log n \text{ heads}) &\leq \binom{a \cdot c \log n}{c \log n} \left(\frac{1}{2}\right)^{(a-1) c \log n} \\
&\leq \frac{(e \cdot a)^{c \log n}}{2^{(a-1) c \log n}} \qquad\qquad \text{Recall: } \binom{y}{x} \leq \left(e \frac{y}{x}\right)^x \\
&= \frac{2^{\log(e \cdot a) c \log n}}{2^{(a-1) c \log n}} \\
&= 2^{\log(e \cdot a) - (a-1)} \\
&= \frac{1}{2^{a-1-log(e \cdot a)}}
\end{aligned}$$

Then the probability that we get more than $c \log n$ heads is $1 - \dfrac{1}{n^\alpha}$ for $\alpha = a - 1 - \log(e \cdot a)$. Therefore, the number of coin flips we need to make until we get $c \log n$ heads is $O(\log n)$, w.h.p.

Remember that the number of moves in a search is at most the number of coin flips until we get $c \log n$ heads. So search costs $O(\log n)$ time, with high probability. $\square$