Announcements

- Skip Lists
- Read chapter 3
Skip Lists

• Problem: searching a linked list takes too long $O(n)$

• Want to search more quickly

• Solution: add more edges that let us skip through items
Skip List

• Consists of a series of lists \( \{S_0, S_1, \ldots, S_h\} \)
• \( S_0 \) contains every item
• For \( i=1, \ldots, h-1 \) \( S_i \) contains a randomly selected subset of \( S_{i-1} \) plus \(-\infty\) and \(+\infty\)
• \( S_h \) contains only \(-\infty\) and \(+\infty\)
Positions

after(p) - position after p on the same level
before(p) - position before p on the same level
below(p) - position below p on the same tower
above(p) - position above p on the same tower
Searching for $k$ in a Skip List

Finds the largest key $\leq k$

Start at top-most level in the left most position $p$
while below($p$)!=$null$ do
  $p<-\text{below}(p)$  //drop down
  while(key(after($p$))$\leq k$ do  
    $p<-\text{after}(p)$  //scan forward
return $p$
Skip List

Search for 7

S₂
-∞ → 5 → +∞

S₁
-∞ → 5 → 9 → +∞

S₀
-∞ → 2 → 5 → 7 → 9 → +∞
Skip List

Search for 7

S0 -inf 2 5 7 9 +inf
S1 -inf 5 9 +inf
S2 -inf 5 +inf

Search for 7
Skip List

Search for 7

S_0: -inf → 2 → 5 → 7 → 9 → +inf
S_1: -inf → 5 → 9 → +inf
S_2: -inf → 5 → +inf
Skip List

Search for 7

$S_0$  
-\(\text{inf}\) \hspace{1cm} \begin{array}{c} \text{2} \\ \text{5} \\ \text{7} \\ \text{9} \end{array} \hspace{1cm} +\text{inf}$

$S_1$  
-\(\text{inf}\) \hspace{1cm} \begin{array}{c} \text{5} \end{array} \hspace{1cm} \begin{array}{c} \text{9} \end{array} \hspace{1cm} +\text{inf}$

$S_2$  
-\(\text{inf}\) \hspace{1cm} \begin{array}{c} \text{5} \end{array} \hspace{1cm} +\text{inf}$
Skip List

Search for 7
Skip List

Search for 7

S_0: -inf - 2 - 5 - 7 - 9 - +inf
S_1: -inf - 5 - 9 - +inf
S_2: -inf - 5 - +inf
Insertion of k

\[ p = \text{Search for } k \text{ using search procedure} \]
\[ \text{Add } k \text{ after item } p \text{ at bottom level} \]
\[ \text{while } \text{random}() < 1/2 \text{ do} \]
\[ \quad \text{while } \text{above}(p) = \text{null} \text{ do} \]
\[ \quad \quad \text{p} \leftarrow \text{before}(p) \]
\[ \quad \text{p} \leftarrow \text{above}(p) \]
\[ \text{insert after item } p \text{ at next higher level} \]
Removal of k

1. Find k
2. Remove k from bottom level
3. Look at next level up
4. If k is present in this level, remove k from this level otherwise exit
5. Goto 3
Cost

- Expected height

Each level has half the expected number of entries as the previous one

\[ P_i \leq n/2^i \]

\[ \Rightarrow \text{Expected number of levels is } O(\log(N)) \]

Book has more formal reasoning
Search Time

- Outer loop executes $O(h)$ which with high likelihood is $O(\log n)$
- Likely to make $O(1)$ operations on given level
- Only considering keys on level $i$ between the current key and the next greater one on level $i+1$
- Half of these keys in the range $[k, \text{next}(k \text{ on level } i+1)]$ should appear in level $i+1$ and only $k$ and next($k$ on level $i+1$)
- Expect to scan small constant number of keys on each level = $O(1)$
- Total search $O(\log n)$
Space Usage

- Bottom level $n$
- Next level $n/2$
- Next level $n/2^2$
- Sum of $n+n/2+n/4... = n(1+1/2+1/4...) = 2n$
- Space = $O(n)$