Lec12 跳跃表

skip lists: (Pugh in 1989)

a new balanced search structure, a data structure that maintains a dynamic set, supporting insertion, deletion and search

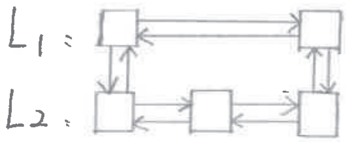
starting from scratch,

a sorted linked list

search takes $\Theta(n)$ time

how can I make it better?

two sorted linked lists, links between equal keys in L_1 and L_2



L_1 stores some subsets

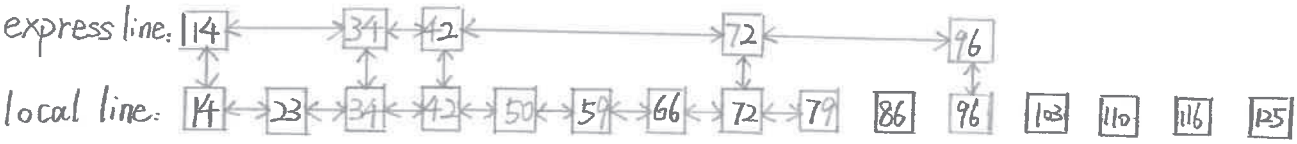
L_2 stores all the elements

Example. (纽约的第七大道线、快线 express lines)

14 23 34 42 50 59 66 72 79 86 96 103 110 116 125

"express and local lines"

站站停车



Search (x)

- walk right in top list L_1 until going right would go to far
- walk down to L_2
- walk right in L_2 until find x or an element $> x$

what keys go in L_1?

best is to spread them out uniformly

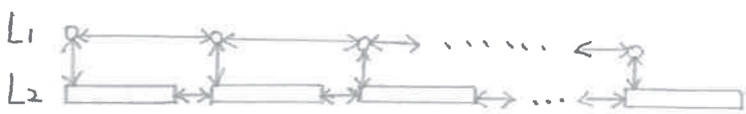
$\Rightarrow$ cost of search $\approx |L_1| + \frac{|L_2|}{|L_1|}$

$\because |L_2|$ 是一个常数 n

$\therefore \min |L_1| + \frac{n}{|L_1|}$

$\Rightarrow |L_1| = \sqrt{n}$

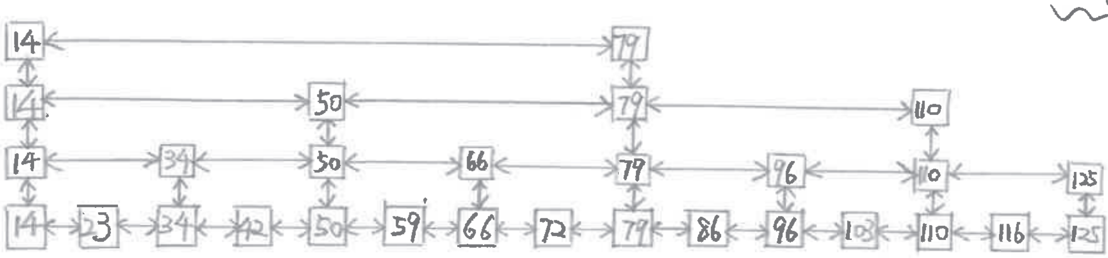
so search cost $\approx 2\sqrt{n}$



3 sorted linked list will cost $3\sqrt[3]{n}$ time for search

k …… $k\sqrt[k]{n}$ ……

$\log_2 n$ …… $\log_2 n \cdot \sqrt[\log_2 n]{n} = \log_2 n \cdot n^{\frac{1}{\log_2 n}} = 2\log_2 n$ ……



诀窍：设 $r = \frac{|L_m|}{|L_{m-1}|}$，共有 x 个 sorted linked list，则 $r^x = n$

若 $x = \log_2 n$，则 $r^{\log_2 n} = n \Rightarrow r = 2$

"like a tree!" "形式上不是树，但逻辑上有点类似"

skip lists maintainance roughly subjects to insert and delete

Insert (x)

- Search (x) to find where x fits in the bottom list
- insert x into the bottom list
- which other lists should store x? (if there are $\log_2 n$ sorted linked lists)

flip a coin, if heads, then promote x to the next level up and flip again

每次都有50%的提升概率

Delete (x)

找到这个元素，把它从出现的链表中一路删除上去。

Theorem:

with high probability, every search in n elements skip lists costs $O(\lg n)$

"w.h.p." 　　　　在有 $\log_2 n$ 级链表的情况下

*define event E occurs w.h.p. if for any $\alpha \geq 1$, there is a suitable choice of constants for which the event E occurs with probability $\geq 1 - O(\frac{1}{n^\alpha})$

失败概率

这里只给出一个引理，其余的证明过程略。

Lemma: w.h.p., #levels $= O(\lg n)$

proof: error probability for $\{\leq c\lg n$ levels$\}$

$= P\{ > c\lg n \text{ levels}\}$

Boole inequality → $\leq n \cdot P\{x \text{ gets promoted} \geq c\lg n \text{ times}\}$

$= n \cdot (\frac{1}{2})^{c\lg n}$

$= n/n^c$

$= 1/n^{c-1} \xlongequal{\text{令 } c-1=\alpha} 1/n^\alpha$ 　Q.E.D.

证明思路：(search backwards)
search starts [ends] at a node in the bottom list, at each node visited,
if the node wasn't promoted higher,
(相当于硬币抛出了反面), then go left,
if the node was promoted, then go up,
and stop [root] (or $-\infty$).