跳 表

问题：

如何在一个有序链表中快速查找元素 k？

例如：

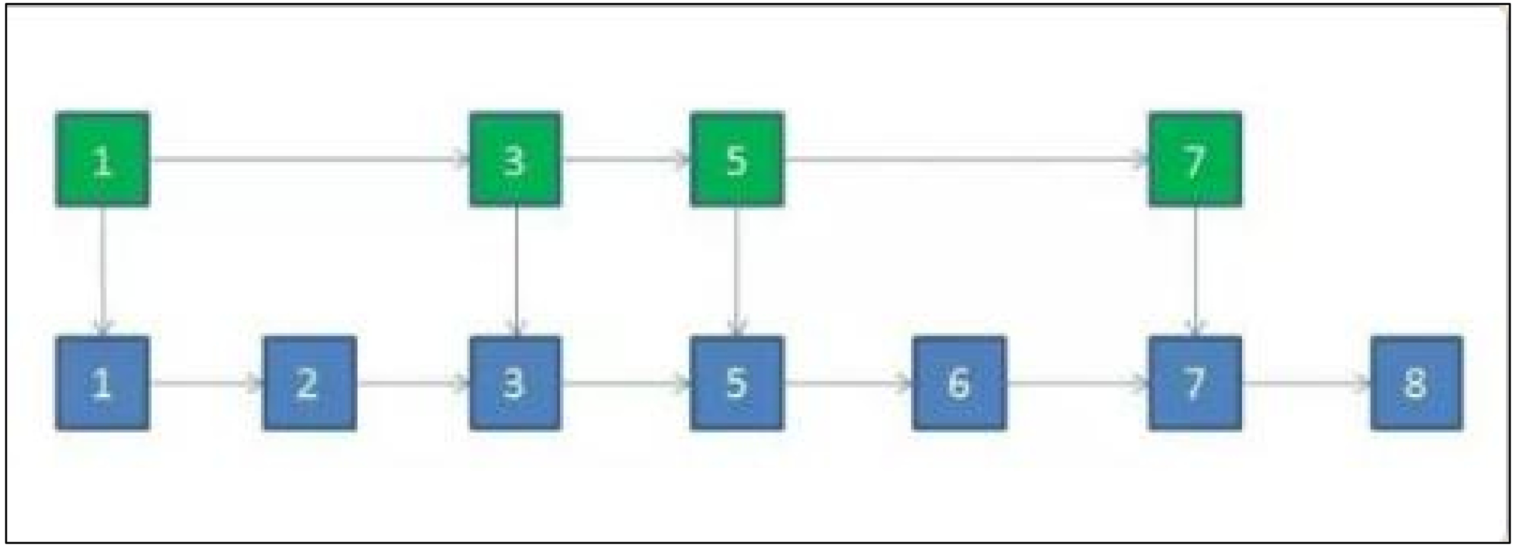
如果要查找 5 ，过程为 1→ 2 → 3 → 5

单链表中查找某个数据的时间复杂度为O(n)，如何更快一些?

跳表

可以这样做：

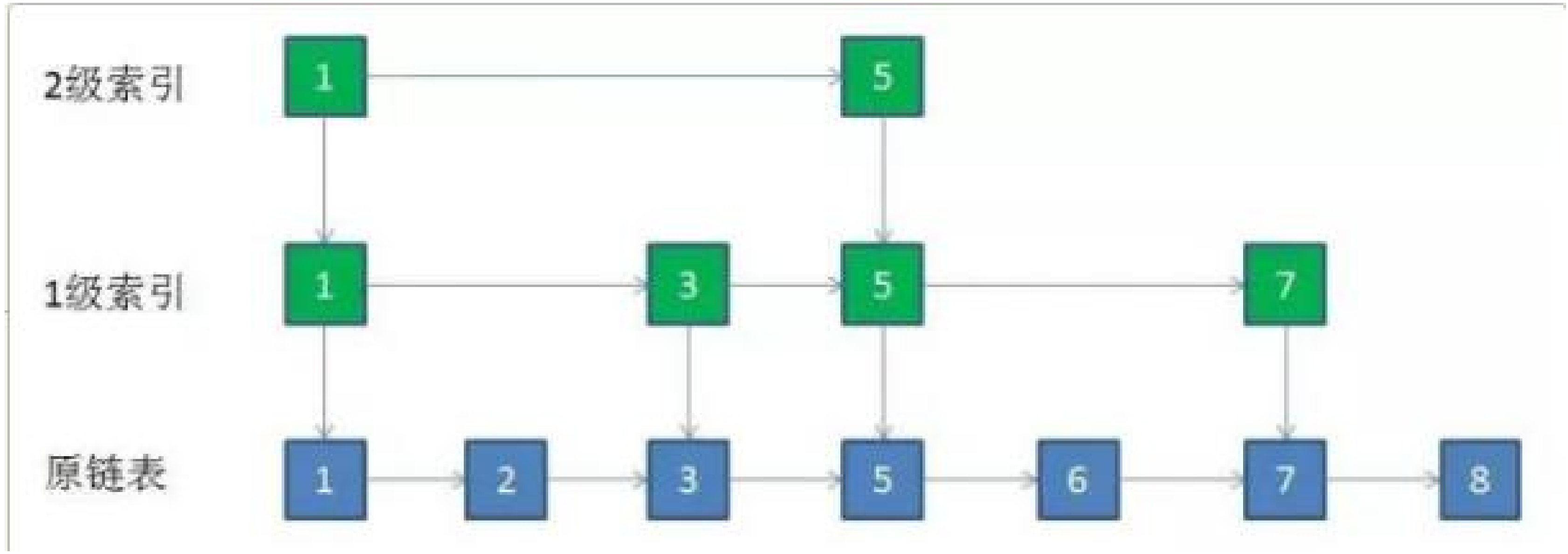
从原链表中取出一些结点再构成一个有序链表，作为原链表的索引。



查找 6，过程为 1→ 3 → 5 → 6。

跳表

跳表，或跳跃表(Skip List)，是一种基于**有序链表**的扩展， 以有序的方式在层次化的链表（多级索引）中保存元素，结构简单，易于实现。



查找 6 ，过程为 1→ 5 → 6。

跳表

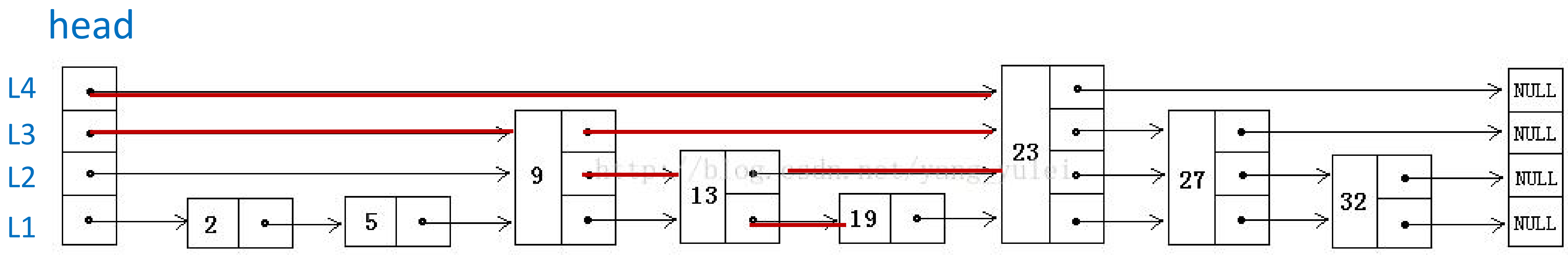
- **特点：**
  - 由多层(至少2层)结构组成,每层都是有序的链表（默认升序）；
  - 最底层包含所有元素；
  - 如果元素x出现在第i层，则所有比i小的层都包含x。
  - 总是从最高层开始访问。

跳表

- 基本操作
  - 创建
  - 查找
  - 插入
  - 删除

跳表

- ## 查找



例如：

查找 19， 过程为 23 → 9→ 13 → 19，查找成功。

查找 28， 过程为 23 → 27，已到最底层，查找失败。

跳表

查找

过程描述：

初始时位于跳表最上层的头结点，然后逐步向右扫描。

（1）若当前元素等于查找元素时，查找成功，退出；

（2）若当前元素大于查找元素时，回退到前一个元素，并下降一层，继续向右扫描。

（3）直到最下层链表，仍未找到待查元素，则查找失败，退出。

跳表

- ## 插入

过程描述：

（1）首先查找跳表，确认新元素应在链表中的插入位置。

（2）然后，在若干层中的适当位置插入新元素。

问题：这里“若干层”如何确定？

跳表是一种随机化数据结构，其随机化体现在插入元素时，元素所占有的层数完全是随机的，层数是通过一个随机函数实现。

跳表

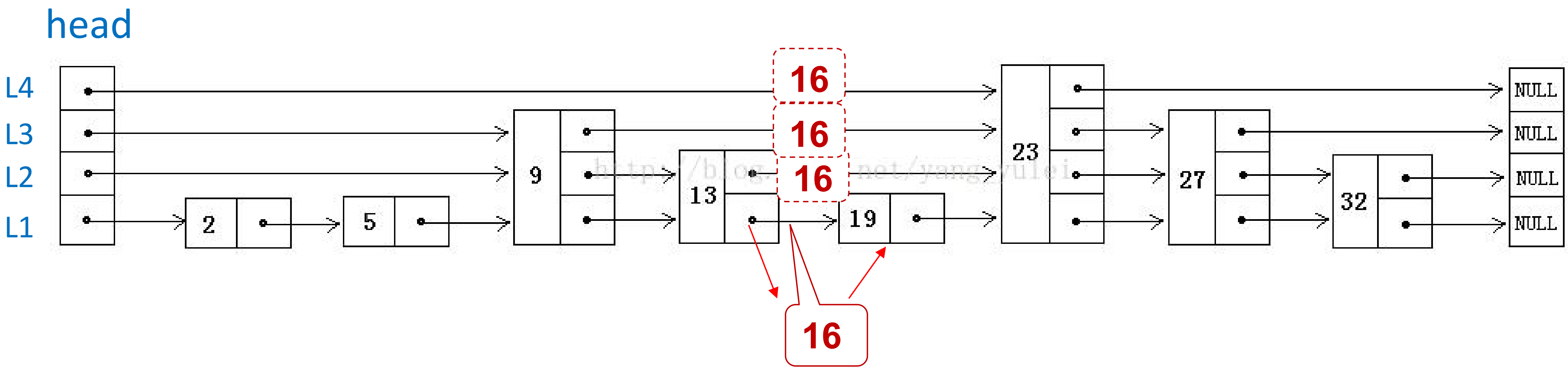
随机化算法：

掷硬币的方式确定应在哪些层插入新元素。

初始层数level设为1（最底层）。调用random( )，返回1（代表'正面'）或0（代表'背面'），每个出现的概率是1/2，如果是1，更新层数level++，否则终止更新。

跳表

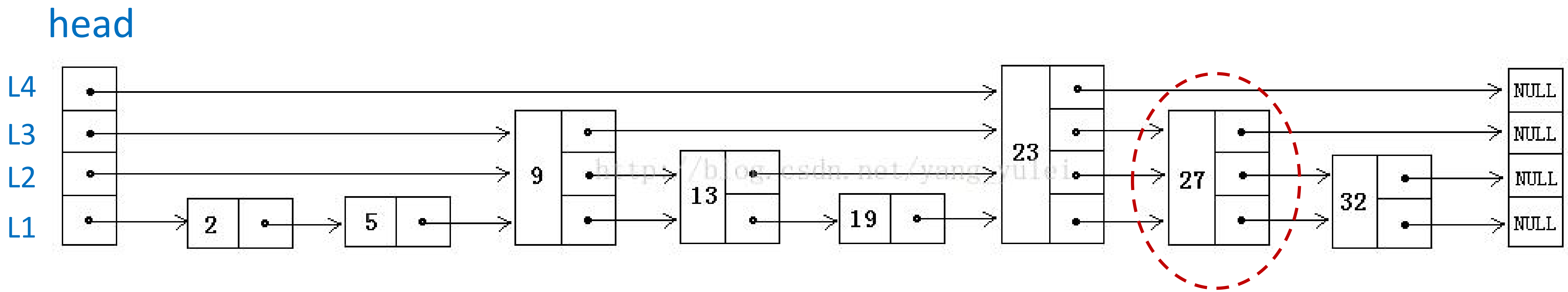
- 插入



插入16。

跳表

- 删除



例如：删除27，过程为 23 → 27，查找到27 ，删除。

跳表

删除

过程描述：

（1）首先查找跳表，从含有关键字为k的元素的最上层链表中删除。

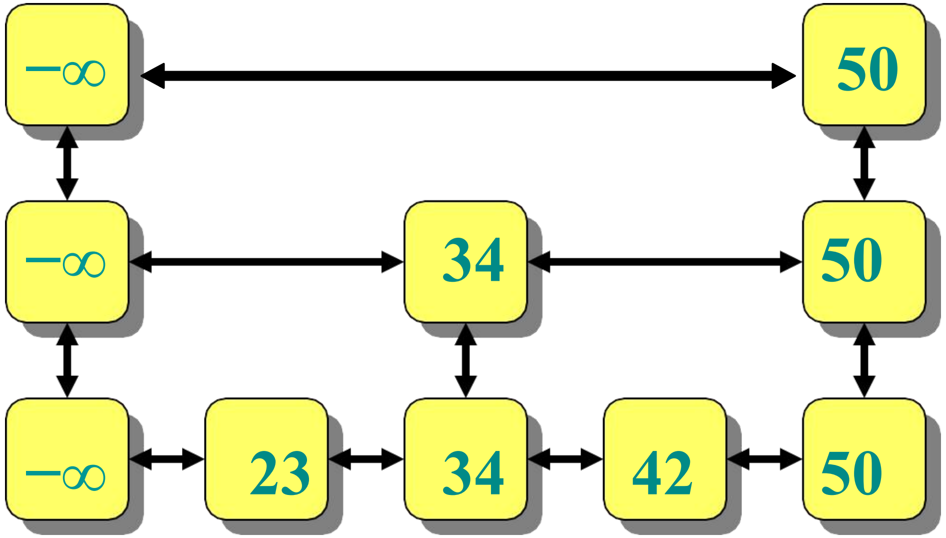
（2）然后，依次删除以下各层中关键字为k的元素，直到最底层。

注：从所有包含该元素的层中删除。

跳表

- 创建

跳表是初始空结构（仅包含$-\infty$）进行系列插入操作后的结果。



跳表

跳表的结点：

down	data	next

或

(假定Maxlevel=7)

data	next[6]
	next[5]
	next[4]
	next[3]
	next[2]
	next[1]
	next[0]

跳表

性能分析

- 时间复杂度 O(logn)

查找操作的时间复杂度与 跳表的索引高度*每层遍历的元素个数 相关。

- 跳表的索引高度

理想情况下每两个结点会抽出一个结点作为上一级索引的结点，原始的链表有n个元素，则第一级索引有n/2个结点，以此类推，k级索引就有n/（2^k)个元素，最高索引一般有两个元素，索引最大高度k= $\log_2 n - 1$，加上原始链表，跳表的总高度k= $\log_2 n$。

- 每层遍历的元素个数

按上面提到的假设，每级索引中都是两个结点抽出一个结点作为上一级索引的结点。从最高一级（只有两个元素）开始，这时的查找就像是在进行二分查找，每一层最多遍历三个元素。

相关证明
(参考)

跳表

性能分析

- 空间复杂度 O(n)

理想情况下，第i层元素个数为 n/(2^i)，所以跳表中元素总数为

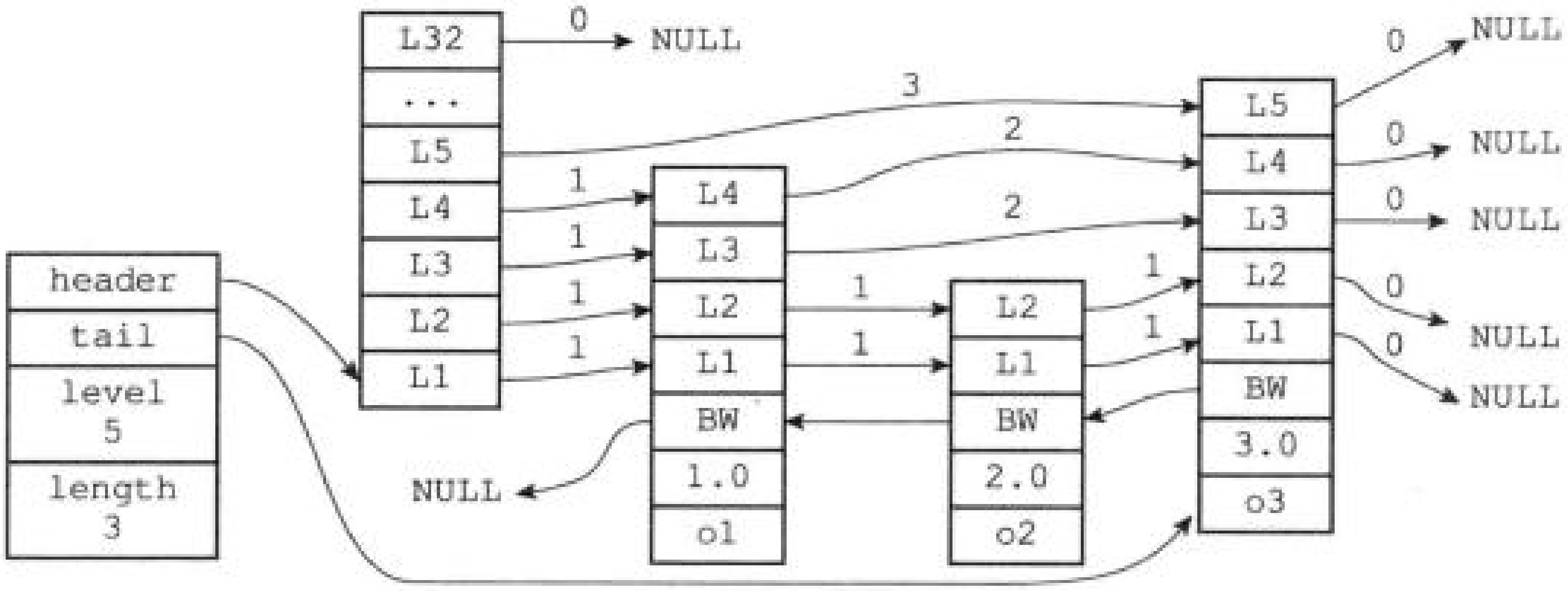
$$\sum_{i=0}^{k} \frac{n}{2^i} < 2n$$

注：跳表是多级索引链表结构，是一种典型的“以空间换取时间”的做法。

跳表

- 应用

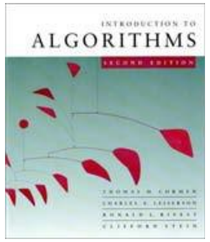
跳跃表在Redis里的用处：一是实现有序集合，另一个是在集群结点中用作内部数据结构。



With-high-probability theorem

THEOREM: *With high probability*, every search in a skip list costs $O(\lg n)$

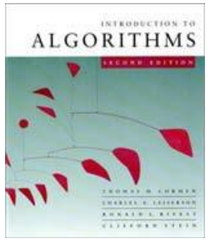
- **INFORMALLY:** Event E occurs ***with high probability*** (***w.h.p.***) if, for any $\alpha \geq 1$, there is an appropriate choice of constants for which E occurs with probability at least $1 - O(1/n^\alpha)$
  - In fact, constant in $O(\lg n)$ depends on α
- **FORMALLY:** Parameterized event E_α occurs ***with high probability*** if, for any $\alpha \geq 1$, there is an appropriate choice of constants for which E_α occurs with probability at least $1 - c_\alpha/n^\alpha$



With-high-probability theorem

THEOREM: With high probability, every search in a skip list costs $O(\lg n)$

- **INFORMALLY:** Event E occurs ***with high probability*** (***w.h.p.***) if, for any $\alpha \geq 1$, there is an appropriate choice of constants for which E occurs with probability at least $1 - O(1/n^{\alpha})$
- **IDEA:** Can make ***error probability*** $O(1/n^{\alpha})$ very small by setting α large, e.g., 100
- Almost certainly, bound remains true for entire execution of polynomial-time algorithm



Boole's inequality / union bound

Recall:

BOOLE'S INEQUALITY / UNION BOUND:

For any random events $E_1, E_2, \ldots, E_k$,

$$\Pr\{E_1 \cup E_2 \cup \ldots \cup E_k\}$$
$$\leq \Pr\{E_1\} + \Pr\{E_2\} + \ldots + \Pr\{E_k\}$$

Application to with-high-probability events:

If $k = n^{O(1)}$, and each E_i occurs with high probability, then so does $E_1 \cap E_2 \cap \ldots \cap E_k$

Analysis Warmup

LEMMA: With high probability, n-element skip list has $O(\lg n)$ levels

PROOF:

- Error probability for having at most $c \lg n$ levels

$$= \Pr\{\text{more than } c \lg n \text{ levels}\}$$

$$\le n \cdot \Pr\{\text{element } x \text{ promoted at least } c \lg n \text{ times}\}$$

(by Boole's Inequality)

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

$$= n \cdot (1/n^c)$$

$$= 1/n^{c-1}$$

Analysis Warmup

LEMMA: With high probability, n-element skip list has $O(\lg n)$ levels

PROOF:

- Error probability for having at most $c \lg n$ levels $\leq 1/n^{c-1}$
- This probability is ***polynomially small***, i.e., at most n^{α} for $\alpha = c - 1$.
- We can make α arbitrarily large by choosing the constant c in the $O(\lg n)$ bound accordingly. □

Proof of theorem

THEOREM: With high probability, every search in an n-element skip list costs $O(\lg n)$

COOL IDEA: Analyze search backwards—leaf to root

- Search starts [ends] at leaf (node in bottom level)
- At each node visited:
  - If node wasn't promoted higher (got TAILS here), then we go [came from] left
  - If node was promoted higher (got HEADS here), then we go [came from] up
- Search stops [starts] at the root (or $-\infty$)

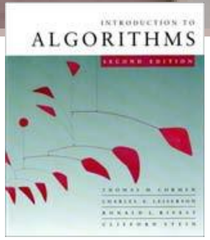
Proof of theorem

THEOREM: With high probability, every search in an n-element skip list costs $O(\lg n)$

COOL IDEA: Analyze search backwards—leaf to root

PROOF:

- Search makes "up" and "left" moves until it reaches the root (or $-\infty$)
- Number of "up" moves < number of levels $\le c \lg n$ w.h.p. *(Lemma)*
- $\Rightarrow$ w.h.p., number of moves is at most the number of times we need to flip a coin to get $c \lg n$ HEADS



Coin flipping analysis

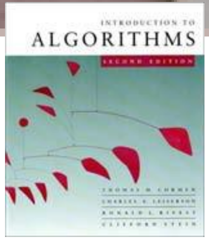
CLAIM: Number of coin flips until $c \lg n$ HEADS $= \Theta(\lg n)$ with high probability

PROOF:

Obviously $\Omega(\lg n)$: at least $c \lg n$

Prove $O(\lg n)$ "by example":

- Say we make $10\, c \lg n$ flips
- When are there at least $c \lg n$ HEADS? (Later generalize to arbitrary values of 10)



Coin flipping analysis

CLAIM: Number of coin flips until $c \lg n$ HEADS $= \Theta(\lg n)$ with high probability

PROOF:

- $\Pr\{\text{exactly } c \lg n \text{ HEADS}\} = \underbrace{\binom{10c\lg n}{c\lg n}}_{\text{orders}} \cdot \underbrace{\left(\frac{1}{2}\right)^{c\lg n}}_{\text{HEADS}} \cdot \underbrace{\left(\frac{1}{2}\right)^{9c\lg n}}_{\text{TAILS}}$

- $\Pr\{\text{at most } c \lg n \text{ HEADS}\} \le \underbrace{\binom{10c\lg n}{c\lg n}}_{\text{overestimate on orders}} \cdot \underbrace{\left(\frac{1}{2}\right)^{9c\lg n}}_{\text{TAILS}}$

Coin flipping analysis (cont'd)

- Recall bounds on $\binom{y}{x}$: $\left(\frac{y}{x}\right)^x \le \binom{y}{x} \le \left(\frac{ey}{x}\right)^x$

- $\Pr\{\text{at most } c \lg n \text{ HEADS}\} \le \binom{10c\lg n}{c\lg n}\left(\frac{1}{2}\right)^{9c\lg n}$

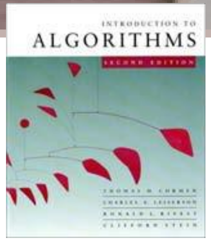
$$\le \left(e\frac{10c\lg n}{c\lg n}\right)^{c\lg n} \cdot \left(\frac{1}{2}\right)^{9c\lg n}$$

$$= (10e)^{c\lg n} 2^{-9c\lg n}$$

$$= 2^{\lg(10e)\cdot c\lg n} 2^{-9c\lg n}$$

$$= 2^{[\lg(10e)-9]\cdot c\lg n}$$

$$= 1/n^{\alpha} \quad \text{for} \quad \alpha = [9-\lg(10e)]\cdot c$$



Coin flipping analysis (cont’ d)

- $\Pr\{$at most $c \lg n$ HEADS$\} \le 1/n^{\alpha}$ for $\alpha = [9 - \lg(10e)]c$
- **KEY PROPERTY:** $\alpha \to \infty$ as $10 \to \infty$, for any c
- So set 10, i.e., constant in $O(\lg n)$ bound, large enough to meet desired α $\square$

This completes the proof of the coin-flipping claim and the proof of the theorem.