布隆过滤器

输入100000个数（小于10^32），每输入一个数，输出这个数是否在之前出现过

强制在线

只允许使用80000个int变量

不进行冲突判别的散列表？

考虑最高效地利用这80000个int，即每一个bit存储一个数是否出现过。

HASH表的大小为80000*32=2560000

那么，正确率？

假设HASH函数映射到元素的概率是随机的

假设输入了一个无重复元素的数据

也就是说，只要数据中存在两个元素的HASH函数结果相同，则出错

正确率：$\sum_{i=1}^{100000}(2560000-i)/2560000 \quad \approx 0$

瞎搞？加补丁？

也许可以，但我没有测试或计算过具体的正确率。

看上去正确率比较高的方法并不是十分简单。

布隆过滤器

布隆过滤器 (Bloom Filter)是由Burton Howard Bloom于1970年提出，它是一种space efficient的概率型数据结构，用于判断一个元素是否在集合中。在垃圾邮件过滤的黑白名单方法、爬虫(Crawler)的网址判重模块中等等经常被用到。哈希表也能用于判断元素是否在集合中，但是布隆过滤器只需要哈希表的1/8或1/4的空间复杂度就能完成同样的问题。布隆过滤器可以插入元素，但不可以删除已有元素。其中的元素越多，false positive rate(误报率)越大，但是false negative (漏报)是不可能的。

算法简介

一个empty bloom filter是一个有m bits的bit array，每一个bit位都初始化为0。并且定义有k个不同的hash function，每个都以uniform random distribution将元素hash到m个不同位置中的一个。在下面的介绍中n为元素数，m为布隆过滤器或哈希表的slot数，k为布隆过滤器重hash function数。

为了add一个元素，用k个hash function将它hash得到bloom filter中k个bit位，将这k个bit位置1。

为了query一个元素，即判断它是否在集合中，用k个hash function将它hash得到k个bit位。若这k bits全为1，则此元素在集合中；若其中任一位不为1，则此元素比不在集合中（因为如果在，则在add时已经把对应的k个bits位置为1）。

优点：

时间高效　　　　容易实现　　　　空间使用少

缺点：

并非绝对的正确率　　不能删除　　不能更改、查询出现次数

正确率？

假设布隆过滤器中的hash function满足simple uniform hashing假设：每个元素都等概率地hash到m个slot中的任何一个，与其它元素被hash到哪个slot无关。

若m为bit数，则对某一特定bit位在一个元素由某特定hash function插入时没有被置位为1的概率为： $1-\frac{1}{m}$

则k个hash function中没有一个对其置位的概率为： $\left(1-\frac{1}{m}\right)^{k}$

如果插入了n个元素，但都未将其置位的概率为： $\left(1-\frac{1}{m}\right)^{kn}$

则此位被置位的概率为： $1-\left(1-\frac{1}{m}\right)^{kn}$

现在考虑query阶段，若对应某个待query元素的k bits全部置位为1，则可判定其在集合中。因此将某元素误判的概率为： $\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)^{k}$

由于 $(1+\mathrm{x})^{\frac{1}{x}} \sim e$,当 $\mathrm{x} \to 0$ 时，并且 $-\frac{1}{m}$ 当m很大时趋近于0，所以

$$\left(1-\left(1-\frac{1}{m}\right)^{kn}\right)^{k}=\left(1-\left(1-\frac{1}{m}\right)^{-m\cdot\frac{-kn}{m}}\right)^{k}\sim\left(1-e^{-\frac{nk}{m}}\right)^{k}$$

现在计算对于给定的m和n，k为何值时可以使得误判率最低。

设误判率为k的函数为：

$$f(k) = \left(1 - e^{-\frac{nk}{m}}\right)^k$$

设 $b = e^{\frac{n}{m}}$ ，则简化为 $f(k) = (1 - b^{-k})^k$ ，

两边取对数 $lnf(k) = k \cdot ln(1 - b^{-k})$

两边对k求导 $$\frac{1}{f(k)} \cdot f'(k) = ln(1 - b^{-k}) + k \cdot \frac{1}{1 - b^{-k}} \cdot (-b^{-k}) \cdot lnb \cdot (-1)$$

$$= ln(1 - b^{-k}) + k \cdot \frac{b^{-k} \cdot lnb}{1 - b^{-k}}$$

下面求最值

$$ln(1-b^{-k})+k\cdot\frac{b^{-k}\cdot lnb}{1-b^{-k}}=0$$

$$\Rightarrow (1-b^{-k})\cdot ln(1-b^{-k})=-k\cdot b^{-k}\cdot lnb$$

$$\Rightarrow (1-b^{-k})\cdot ln(1-b^{-k})=b^{-k}\cdot lnb^{-k}$$

$$\Rightarrow 1-b^{-k}=b^{-k}$$

$$\Rightarrow b^{-k}=\frac{1}{2}$$

$$\Rightarrow e^{-\frac{kn}{m}}=\frac{1}{2}$$

$$\Rightarrow \frac{kn}{m}=ln2$$

$$\Rightarrow k=ln2\cdot\frac{m}{n}=0.7\cdot\frac{m}{n}$$

因此，即当 $k = 0.7 \cdot \frac{m}{n}$ 时误判率最低，此时误判率为：

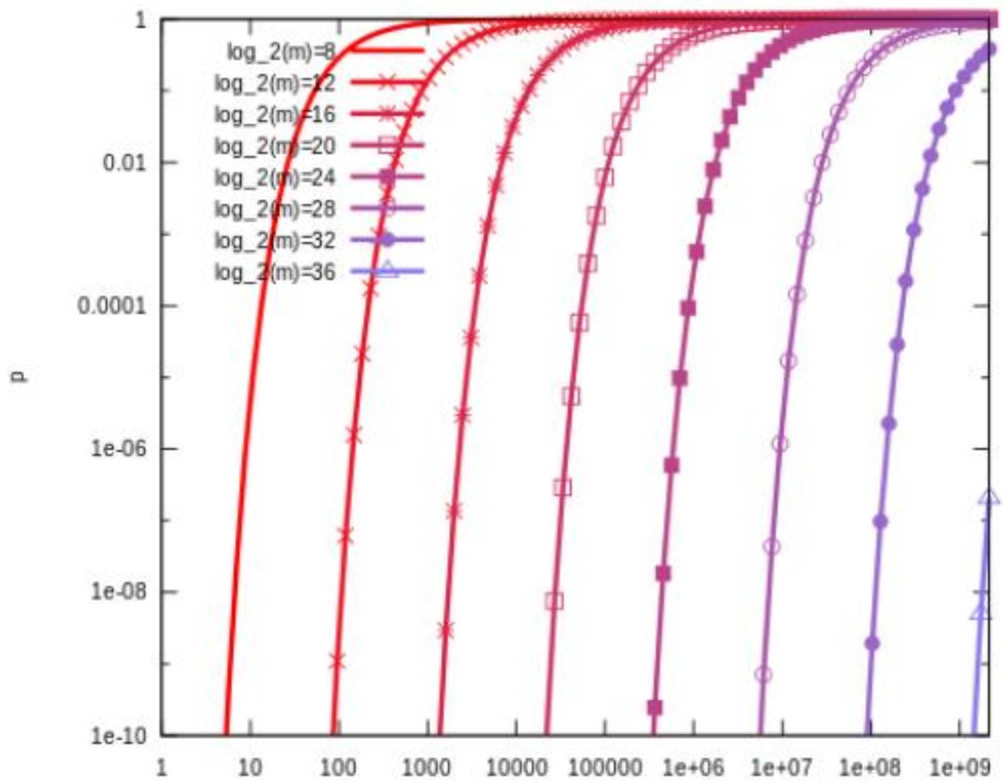
$$P(error) = \left(1 - \frac{1}{2}\right)^k = 2^{-k} = 2^{-ln2 \cdot \frac{m}{n}} \approx 0.6185^{\frac{m}{n}}$$

可以看出若要使得误判率≤1/2，则：

$$\mathrm{k} \geq 1 \Rightarrow \frac{m}{n} \geq \frac{1}{ln2}$$

这说明了若想保持某固定误判率不变，布隆过滤器的bit数m与被add的元素数n应该是线性同步增加的。

下图是布隆过滤器假正例概率 p 与位数组大小 m 和集合中插入元素个数 n 的关系图，假定 Hash 函数个数选取最优数目：$k = (m/n) \ln 2$



回到之前的问题

m/n=2560000/100000=25.6>25

k=25*ln 2≈17

最大单点错误率： $f(k) = \left(1 - e^{-\frac{nk}{m}}\right)^k$ ≈0.0000045848

100000个点的正确率≈96.6%

可以看出，虽然并不是绝对正确，布隆过滤器依旧是很优秀的