一种过滤器,用于判断一个元素在一个集合中是否存在.
它实际上是一个很长的二进制矢量和一系列随机映射函数。布隆过滤器可以用于检索一个元素是否在一个集合中。它的优点是空间效率和查询时间都远远超过一般的算法，缺点是有一定的误识别率和删除困难。

来自〈https://blog.csdn.net/iam333/article/details/38084137〉

原理:

一个元素通过K个不同的hash函数随机散列到bit数组的K个位置上，

Bloom Filter的一个例子集合S｛x，y，z｝。带有颜色的箭头表示元素经过k（k＝3）hash函数的到在M（bit数组）中的位置。元素W不在S集合中，因为元素W经过k个hash函数得到在M（bit数组）的k个位置中存在值为0的位置。

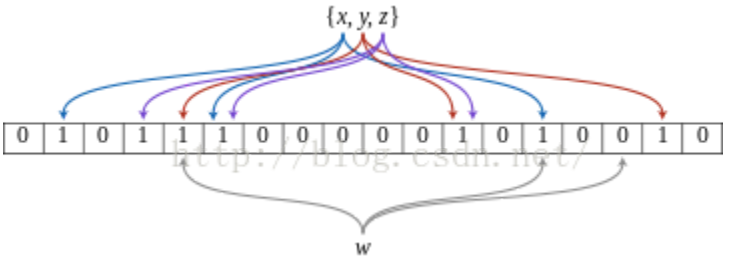
向集合S中添加元素x：x经过k个散列函数后，在M中得到k个位置，然后，将这k个位置的值设置为1。

判断x元素是否在集合S中：x经过k个散列函数后，的到k个位置的值，如果这k个值中间存在为0的，说明元素x不在集合中——元素x曾经插入到过集合S，则M中的k个位置会全部置为1；如果M中的k个位置全为1，则有两种情形。

情形一：这个元素在这个集合中；

情形二：曾经有元素插入的时候将这k个位置的值置为1了（第一类错误产生的原因FalsePositive）

来自〈https://blog.csdn.net/lvsaixia/article/details/51503231〉



来自〈https://blog.csdn.net/lvsaixia/article/details/51503231〉

总结:

优点

1. 存储空间和插入/查询时间都是常数，远远超过一般的算法
2. Hash函数相互之间没有关系，方便由硬件并行实现
3. 不需要存储元素本身，在某些对保密要求非常严格的场合有优势

缺点

1. 有一定的误识别率
2. 删除困难

来自〈https://www.cnblogs.com/Jack47/p/bloom_filter_intro.html〉

特性:

过滤器说不存在,那这个元素一定不存在;

过滤器说存在,这个元素可能存在(有错误率)

1.为什么有错误率?

由原理决定的,一个元素被映射成多个值,需比较的值也会被映射成多个值,这个两个元素映射后的值可能会有重合,导致误判,但是需比较映射的值有个为0,则该值一定不存在.也可以说,你的集合越大,误判的次数就越大,按数学推理来算大约是0.0005

2.为什么删除困难?

映射后的值可能有冲突,你删除一个元素,就必须连同映射后的值一起删除,这就可能影响到其他值(也用到了这个映射值),而且布隆本身就存在误判,万一你要删除的它说不存在呢(实际上存在的)