

Bloom filters

Arpita Korwar

May 6, 2010

1 Introduction

Bloom filters are used for answering queries on set membership. In this data structure, the whole element is not stored at the hashed address. Only a few bits are set in an array.

Given a set S of cardinality n , we store it in an array of m bits using k hash functions $h_1(), \dots, h_k()$. Initially, all the cells in the array are set to 0. Then, for each element in the set, $x \in S$, for each $1 \leq i \leq k$, we set $h_i(x) = 1$. If an element is already 1, then it is not modified. To *lookup* an element z , the k locations $h_1(z), h_2(z), \dots, h_k(z)$ are checked. If atleast one of the locations has a 0, then z does not belong to the set and **false** is returned. If all the locations contain a 1, then **true** is returned. But the element may still not belong to the set S . This situation is known as *false positive*.

What about the universality of the hash functions? The hash functions need to be $(n+1)k$ -universal. The hash addresses of the elements in the set and one element that is looked up should be independent.

Given a constant bits per element ratio $\frac{m}{n} = c$, we would like to reduce the probability of a false positive. It can be shown that the minimum probability of false positive occurs when the fraction of 0's in the array is $\frac{1}{2}$ and the number of hash functions $k = \frac{m}{n} \ln 2$. Then, the probability of a false positive is $(0.6185)^{\frac{m}{n}}$. The analysis for this can be found in the book by Mitzenmacher [MU05, Section 5.5.3]. This allows for a tradeoff between the space efficiency and the probability of a false positive.

2 Less hashing, same performance [KM06]

The above specification of Bloom filters uses k hash functions that are $(n+1)k$ -universal. Can we obtain the same features - constant probability

of a false positive with constant bits per element - using fewer hash functions? The answer is yes, we can obtain the same features with only 2 hash functions.

We build k hash functions $g_i, 1 \leq i \leq k$ from 2 hash functions h_1 and h_2 that are picked uniformly and randomly. Any element x is hashed with $g_i(x) = h_1(x) + ih_2(x)$. It has been shown that this hashing scheme gives the same probability of false positive as the classic bloom filter.

2.1 k separate address spaces

In this subsection, we will study a special case of the hashing scheme described above.

The hash functions h_1 and h_2 are both picked independently. Presently we will assume that these functions are picked from the family of *totally random hash functions* from the universe U to $\{0, \dots, p-1\}$, p a prime¹. The array of address space bits is divided into k subarrays, each of range $\{0, \dots, p-1\}$. The size of the array is $m = kp$. Initially, all the bits in this array are set to 0. Then each element is hashed according to g_0, \dots, g_{k-1} , given by $g_i(x) = h_1(x) + ih_2(x) \pmod p$. Henceforth, in this section, i will be a value in $\{0, 1, \dots, k-1\}$. On input x , the bit $g_i(x)$ of the i th subarray is set to 1. Hence, when considering collisions between elements x and y ($x \neq y$), we only need to consider the collisions that occur on the same hash function, i.e. $g_i(x) = g_i(y)$. To lookup an element z , like in the classic Bloom filters, the bit at $g_i(z)$ in the i th subarray is checked. If all these bits have value 1, we conclude that $z \in S$. But these bits may have been set by some other elements. This leads to false positive. We would analyze this scheme to find the probability of false positive. Let us first study a useful behavior of this scheme.

Claim 1. *If $x \neq y$, then their hash addresses $g_i(x), g_i(y)$ behave only in one of the following manners:*

1. $\forall i, g_i(x) = g_i(y)$ iff $h_1(x) = h_1(y)$ and $h_2(x) = h_2(y)$.
2. $g_i(x) = g_i(y)$ for exactly one i , given by $i = (h_1(y) - h_1(x))(h_2(x) - h_2(y))^{-1} \pmod p$ ($h_2(x) \neq h_2(y)$).
3. $\forall i, g_i(x) \neq g_i(y)$.

Proof. This claim implies that it will never happen that $g_i(x) = g_i(y)$ for exactly 2 values of i (or $3, 4, \dots, k-1$ values of i).

¹Later we will show that the family of hash functions needs to be only $n+1$ -universal.

We consider the following four cases:

I $h_1(x) = h_1(y)$ and $h_2(x) = h_2(y)$.

II $h_1(x) = h_1(y)$ and $h_2(x) \neq h_2(y)$.

III $h_1(x) \neq h_1(y)$ and $h_2(x) = h_2(y)$.

IV $h_1(x) \neq h_1(y)$ and $h_2(x) \neq h_2(y)$.

When case I occurs, $\forall i, g_i(x) = g_i(y)$.

When case III occurs, definitely $g_0(x) \neq g_0(y)$. Also the distance between the values of $g_i(x)$ and $g_i(y)$ is maintained in subsequent values of i .

Let $h_1(x) - h_1(y) = d_1 \pmod p$ and $h_2(x) - h_2(y) = d_2 \pmod p$. Then $g_i(x) - g_i(y) = d_1 + id_2 \pmod p$. Since the range $\{0, 1, \dots, p-1\}$ forms the field \mathbb{F}_p , we can perform field operations like addition, division, etc. If $d_1 = d_2 = 0$, then this equation holds for all values of i (i.e. case I). When $d_2 \neq 0$ (cases II and IV) we get $i = -d_1 d_2^{-1} \pmod p$. We get a unique value for i . If this i lies in the range $\{0, 1, \dots, k-1\}$, then the collision of x and y occurs exactly at the i th subarray. Else, no collision occurs. \square

Let us now proceed to find the probability of false positive. We call the event of false positive as event \mathcal{F} . Let S be a set of cardinality n that is stored in the Bloom filter. Let $z \notin S$. Let us also fix $h_1(z)$ and $h_2(z)$. A false positive occurs when for every i th bucket, some element in the subarray hashes to the same cell as z . One way this may occur is when $\exists x \in S : \forall i g_i(x) = g_i(z)$. Let us call this event \mathcal{E} .

As can be seen from claim 1,

$$\begin{aligned}
\Pr[\mathcal{E}] &= \Pr\left[\exists x \in S : \left(h_1(x) = h_1(y) \wedge h_2(x) = h_2(y)\right)\right] \\
&= 1 - \Pr\left[\forall x \in S : \neg\left(h_1(x) = h_1(y) \wedge h_2(x) = h_2(y)\right)\right] \\
&= 1 - \left(\Pr\left[\neg\left(h_1(x) = h_1(y) \wedge h_2(x) = h_2(y)\right)\right]\right)^n \\
&= 1 - \left(1 - \Pr\left[h_1(x) = h_1(y) \wedge h_2(x) = h_2(y)\right]\right)^n \\
&= 1 - \left(1 - \frac{1}{p^2}\right)^n \\
&= 1 - \left(1 - \frac{k^2}{m^2}\right)^n, \text{ (since } m = kp\text{)} \\
&\approx 1 - e^{-\frac{kn}{m^2}}
\end{aligned}$$

When we substitute the optimal value for k that was found in the analysis of classical Bloom filters, i.e. $k = \frac{m}{n} \ln 2$, we get $\Pr[\mathcal{E}] = 1 - e^{-\frac{c_1}{n}}$ (c_1 is a constant). Hence, $\Pr[\mathcal{E}] = o(1)$.

Now, probability of false positive,

$$\begin{aligned}\Pr[\mathcal{F}] &= \Pr[\mathcal{F}|\mathcal{E}] \cdot \Pr[\mathcal{E}] + \Pr[\mathcal{F}|\neg\mathcal{E}] \cdot \Pr[\neg\mathcal{E}] \\ &= o(1) + \Pr[\mathcal{F}|\neg\mathcal{E}] (1 - o(1)).\end{aligned}$$

Hence, probability of false positive can be calculated once we bound $\Pr[\mathcal{F}|\neg\mathcal{E}]$.

2.1.1 Finding $\Pr[\mathcal{F}|\neg\mathcal{E}]$

We have to find the probability that there is no x such that $g_i(x) = g_i(z)$, but for each i , $\exists x_i \in S$ such that $g_i(x_i) = g_i(z)$ in exactly one place (from claim 1). Let us first fix the size of the sample space. It can be seen that the value of d_1 in the proof of claim 1 depends on values of $h_1(x)$ and $h_1(z)$. The value of d_2 depends on values of $h_2(x)$ and $h_2(z)$. The values of $h_1(z)$ and $h_2(z)$ are fixed. What values can $(h_1(x), h_2(x))$ take? Since event \mathcal{E} cannot occur, there are $p^2 - 1$ possible values for $(h_1(x), h_2(x))$. This is the size of the sample space.

In how many such values for $(h_1(x), h_2(x))$ does $x \in S$ collide with z in one of the k buckets? Probability of x colliding with z is the same in all buckets. For bucket number i^* , once d_2 ($\neq 0$) is fixed, $d_1 = -id_2 \pmod p$ becomes fixed. Hence for each value of i^* there are $p - 1$ pairs $(h_1(x), h_2(x))$ such that x collides with z in the i^* th bucket. Hence probability that x collides with z in one of the k buckets is $\frac{k(p-1)}{p^2-1} = \frac{k}{p+1}$.

We can formulate the problem of finding $\Pr[\mathcal{F}|\neg\mathcal{E}]$ as a balls and bins problem. There are n balls and k bins. Each ball falls into the set of bins with probability $\frac{k}{p+1}$. If it does not fall into any bin, it is discarded. If it does fall in a bin, then it falls in one of the k bins uniformly and randomly. We have to find the probability that none of the k bins are empty.

Binomial variable:[MU05, Section 2.2] Suppose we have to perform n independent experiments where the probability of success is p and the random variable X gives the number of successes. The distribution of the r.v. X is given by

$$\Pr [X = j] = \binom{n}{j} p^j (1 - p)^{n-j}.$$

The expected value of X , $E[X] = np$.

Poisson distribution:[MU05, Section 5.3] In the above situation, when n grows larger and larger, we can use the Poisson distribution to find $\Pr [X = j]$. If the expected number of successes in n independent experiments is μ then $\Pr [X = j] = \frac{e^{-\mu} \mu^j}{j!}$.

If a ball falls in a bin with probability p , then the distribution of number of balls in that bin when n balls are thrown follows Poisson distribution with parameter $\mu = np$. This experiment can be extended to m bins in two ways:

1. n balls are thrown into the m bins; the probability of a ball falling into a bin is p . Then the number of balls in each bin is not independent, since the total should add up to n . Let us call this exact experiment.
2. Each bin has n balls thrown at it, such that they fall into the bin with probability p . The number of balls in each bin in this experiment are independent of the number of balls in other bins. Let us call this the independent experiment.

It can be proved that the probability of any event in the exact experiment and the probability of that event in the independent experiment are approximately the same (refer [MU05, Corollary 5.9]).

The experiment of the n balls falling in one of the bins with probability $\frac{k}{p+1}$ follows a binomial distribution with number of experiments = k and probability of success = $\frac{k}{p+1} < \frac{k}{p}$. The expected number of successes is $\frac{kn}{p} = \frac{k^2 n}{m} = \frac{k^2}{c}$ (The number of bits per element is constant, $\frac{m}{n} = c$). The total number of balls in k bins is given by binomial distribution with parameters $\left(n, \frac{k}{p}\right)$ is equivalent to Poisson distribution with parameter $\mu = \frac{k^2}{c}$.

If the ball falls in the set of bins, it can fall only in any one of the bins. Each bin is selected with equal probability. This is equivalent to the exact experiment. We know that even if we replace this exact experiment with independent experiment, the probability of all bins getting at least 1 ball is

the same. Hence, we analyze the exact experiment where n balls are thrown into each of the k bins and probability that a ball falls into the bin is $\frac{k}{c}$ (so that with k bins, the probability of a ball falling in one of the bins is $\frac{k^2}{c}$).

$$\begin{aligned} \Pr[\text{bin is empty}] &= \frac{e^{-\mu} \mu^0}{0!} = e^{-\frac{k}{c}} \\ \Pr[\text{bin is non-empty}] &= 1 - e^{-\frac{k}{c}} \\ \Pr[\text{all bins are non-empty}] &= \left(1 - e^{-\frac{k}{c}}\right)^k \end{aligned}$$

Hence, $\Pr[\mathcal{F}|\neg\mathcal{E}] = \left(1 - e^{-\frac{k}{c}}\right)^k$. Hence $\Pr[\mathcal{F}] = o(1) + \left(1 - e^{-\frac{k}{c}}\right)^k$. $(1 - o(1)) \approx \left(1 - e^{-\frac{k}{c}}\right)^k$. We get the same probability of false positive as in the case of Bloom filters.

2.2 How much independence is required?

We have assumed throughout the section that the family of totally random hash functions is used. But since we are inserting only n elements in the set and since we have partitioned the array into buckets, we can consider each hash function g_i separately. We thus need independence of the n elements in the set and any element that may be checked for set membership. Hence, we need $n + 1$ -universal family of hash functions.

References

- [KM06] Adam Kirsch and Michael Mitzenmacher. Less hashing, same performance: Building a better bloom filter, <http://www.eecs.harvard.edu/~kirsch/pubs/bbbf/esa06.pdf>. *Algorithms ESA, 14th Annual European Symposium*, 2006.
- [MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing*. Cambridge University Press, December 2005.