Tutorial 6 Solution

Introduction

For an overview of the basic ideas and background on Bloom filter see its wikipedia entry or the original paper due to Bloom[1]. Here we only enumerate the relevant results required to solve the tutorial problems.

A Bloom filter can be used to answer queries to test the membership of elements of a set X in the form "does $x \in X$ belong to the subset $S \subseteq X$?". The filter consists of an array of M bits and a collection of K hash functions $f_1, f_2, \ldots f_K : X \mapsto \{0, 1, \ldots M-1\}$. The hash functions are chosen in such a way that (a priori) for each $x \in X$ and each $1 \leq k \leq K$ the probability of mapping x to any of the M bits equals $1/M$.

Initially, all bits in the array are set to zero (which corresponds to $S = \emptyset$). Whenever a new element $x \in X$ is inserted to the filter (corresponding to the operation $S \leftarrow S \cup \{x\}$), one applies all K hash functions to x and sets the bits with indices $f_1(x), f_2(x), \ldots f_K(x)$ to value 1 (true). Note that in the case of a traditional Bloom filter it is not possible to remove elements.

Whenever the filter receives the query "$x \in S$?" one computes again all values $f_1(x), f_2(x), \ldots f_K(x)$ and reads the value of the corresponding bits in the array. If at least one of these bits (still) has a value 0 (false), the filter returns "$x \notin S$". Note that this answer is guaranteed to be correct, since if it was the case that x was previously added to S, then all of these bits would have been set to 1 (true). However, in the case that all of them are set to 1, then the filter returns "$x \in S$", which may be an incorrect answer. If the filter claims that an element is part of S while in fact it is not, then we speak of a *false positive*.

For a given Bloom filter, we are able to obtain the *false-positive probability* based on the assumptions on the size of the array and the number of uniformness properties of the hash functions.

$$
\begin{aligned}
\text{Prob(bit } b \text{ is set to 1 by hash function } f_k) = &\quad \tfrac{1}{M} \\
\text{Prob(bit } b \text{ is not set to 1 by any hash function } f_k) = &\quad (1 - \tfrac{1}{M})^K \\
\text{Prob(bit } b \text{ is still set to 0 after inserting } N \text{ elements}) = &\quad (1 - \tfrac{1}{M})^{KN}
\end{aligned}
$$

Then the false-positive probability can be computed as follows.

$$
\begin{aligned}
& \text{Prob(falsely classifying } x \in S \text{ after inserting } N \text{ elements)} \\
= \;& \text{Prob(all of the } K \text{ bits } f_1(x), f_2(x), \ldots f_K(x) \text{ are set to 1 after inserting } N \text{ elements)} \\
= \;& (1 - \text{Prob(bit } b = f_k(x) \text{ is still set to 0 after inserting } N \text{ elements)})^K \\
= \;& (1 - (1 - \tfrac{1}{M})^{KN})^K \approx (1 - e^{-KN/M})^K \qquad (1)
\end{aligned}
$$

For a fixed array size M and a fixed number of insertions N, one can ask for the (smallest) value of K that minimizes the false-positive probability. Here we only mention the closed form expression for the optimal K, skip the details and instead refer to the literature. So the smallest value of K (denote K^*) that minimizes the false positive probability can be computed as

$$K^* = \frac{M}{N} \ln 2 \approx 0.7 \frac{M}{N}. \qquad (2)$$

Assuming one chooses the number of hash functions according to K^*, one obtains from Equations (1) and (2) a false positive probability of approximately

$$(1 - e^{-K^* N/M})^{K^*} = 2^{-\frac{M}{N}\ln 2}. \tag{3}$$

Solutions

1. a) We have $K = 2$, $M = 10^6 \cdot 8$ bits, $N = 10^5$. From (1) we obtain

   $$\text{Prob(falsely classifying } x \in S \text{ after inserting } N \text{ elements)} \approx (1 - e^{-1/40})^2 \approx 0.00061.$$

   b) Now instead we have $N = 10^6$ and thus obtain from (1)

   $$\text{Prob(falsely classifying } x \in S \text{ after inserting } N \text{ elements)} \approx (1 - e^{-1/4})^2 \approx 0.04893.$$

2. a) We have $M = 10^6 \cdot 8$ bits and $N = 10^6$. From (2) we obtain[1]

   $$K^* \approx 5.6$$

   If we substitute for M and N we obtain from (3)

   $$\begin{aligned} &\text{Prob(falsely classifying } x \in S \text{ after inserting } N \text{ elements)} \\ = \; & 2^{-M/N \cdot \ln 2} \approx 0.021. \end{aligned}$$

   b) Instead we now have $M = 512 \cdot 10^3 \cdot 8$ bits. As before, we use (2) to obtain

   $$K^* \approx 2.9$$

   and for the false-positive probability

   $$\begin{aligned} &\text{Prob(falsely classifying } x \in S \text{ after inserting } N \text{ elements)} \\ = \; & 2^{-M/N \cdot \ln 2} \approx 0.140. \end{aligned}$$

References

[1] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13:422–426, July 1970.

[1] Of course in reality K^* would need to be an integer but let's ignore this here.