Towards Weighted Bloom Filters

The Bloom filter [1] is a data structure that was introduced in 1970 and that has been adopted by the networking research community in the past decade [2] thanks to the bandwidth efficiencies that it offers for the transmission of set membership information between networked hosts. A sender encodes the information into a bit vector, the Bloom filter, that is more compact than a conventional representation. Computation and space costs for construction are linear in the number of elements. The receiver uses the filter to test whether various elements are members of the set. Though the filter will occasionally return a false positive, it will never return a false negative. When creating the filter, the sender can choose its desired point in a trade-off between the false positive rate and the size. The compressed Bloom filter, an extension proposed by Mitzenmacher [3], allows further bandwidth savings.

A Bloom filter is a vector $v$ of $m$ bits that codes the membership of a subset $A=\{a_1, a_2, ..., a_n\}$ of $n$ elements of a universe $U$ consisting of $N$ elements. In most papers, the size of the universe is not specified. However, Bloom filters are only useful if the size of $U$ is much larger than the size of $A$. The idea is to initialize this vector $v$ to 0, and then take a set $H=\{h_1, h_2, ..., h_k\}$ of $k$ independent hash functions $h_1, h_2, ..., h_k$ each with range $\{1, ..., m\}$. For each element $a \in A$, the bits at positions $h_1(a), h_2(a), ..., h_k(a)$ in $v$ are set to 1.

To check if an element $b$ of the universe $U$ belongs to the set $A$, all one has to do is check that the $k$ bits at positions $h_1(b), h_2(b), ..., h_k(b)$ are all set to 1. If at least one bit is set to 0, we are sure that $b$ does not belong to $A$. If all bits are set to 1, $b$ possibly belongs to $A$. There is always a probability that $b$ does not belong to $A$. In other words, there is a risk of false positives. The false positive rate can be evaluated a priori and is a measure of a Bloom filter performance.

Beyond this, Bloom filter comes with a strong assumption. Every element $x \in U$ has exactly the same probability to be tested for membership in the filter. However, it is very frequent, in the real world, that elements to be queried do not have the same probability to be encountered. Some elements may be more frequent than others. If a such a frequent element triggers a false positive, it may be a concern for the filter. A few solutions [4,5,6] have been proposed to deal with this, but, up to now, none of them is acceptable.

In this master thesis, you will have to investigate how a weight can be added to elements belonging to $U$ and how this weight can be part of the filter construction, membership test, and false positive (possibly a false negative) rate computation.

This thesis may also be done as an internship at the Université de Haute-Savoie, France, under the supervision of Prof. Kavé Salamtian.¹

**Contact:** Benoit Donnet (benoit.donnet@ulg.ac.be)

**Références**


¹ See http://www.polytech.univ-savoie.fr/index.php?id=listic-kave-salamatian&L=0


