Network Programming Laboratory

5 June 2023

Exercise

A *Bloom filter* is a probabilistic data structure used to test whether an element is a member of a set $\mathcal{S}$. It uses a bit array of size N and a set of K hash functions to store and retrieve data efficiently. When an element is inserted into the filter, the hash functions generate a set of indexes in the bit array, and those indexes are marked as "true." To check if an element is in the set, the hash functions are again applied, and if any of the corresponding indexes in the bit array are "false," the element is determined to be not in the set. However, there is a small probability of false positives, where the filter may mistakenly report an element as being in the set when it is not. Bloom filters are commonly used in applications where approximate answers or fast membership queries are acceptable, such as caching, spell checking, and network routers.

Hash functions. A simple way of producing K hash functions is to use two hash functions $f(x)$ and $g(x)$ and combine them linearly, as:

$$h_k(x) = f(x) + k \cdot g(x) \pmod{N}, \quad k = 0, 1, \ldots, K-1$$

From the theory, if n elements belong to the set $\mathcal{S}$ and N is the size of the Bloom filter, then the *optimal* value of K is:

$$K = \frac{N}{n} \ln(2).$$

Bloom filter C++ class. Write a C++ class (or structure) to implement a Bloom filter of size N, equipped with K hash functions. The class must implement the `insert(x)` method to add the element `x` to the set, and the `query(x)` method which returns `true` if the element `x` belongs to the set (`false`, if not).
Hint: Define the class template over the values N and K. Which internal member would you use?
Use the plain integer representation of IP addresses. Although not very efficient and well performing, the functions $f(x)$ and $g(x)$ can be simply selected as the *modulo* N integer representation of the IP address and any of its rearrangement (e.g., by swapping the sixteen upper and lower digits).

An approximate IP counter application. We want to use a Bloom filter to obtain an approximate count of the number of *different* IP destinations found in the provided pcap trace. To do so, write a C++ application in which you are requested to:

1. dimension a Bloom filter for a set of around 5K elements;
2. go through the packet trace and insert new IP addresses in the BF;
3. increment the counter only if the IP destination is new;
4. at the end, print the estimated count and compare it with the exact one (how do you get this value?)