Data Structures and Algorithms (INFO-F413)
Assignment 1: Bloom Filters

Jean Cardinal

October 2025

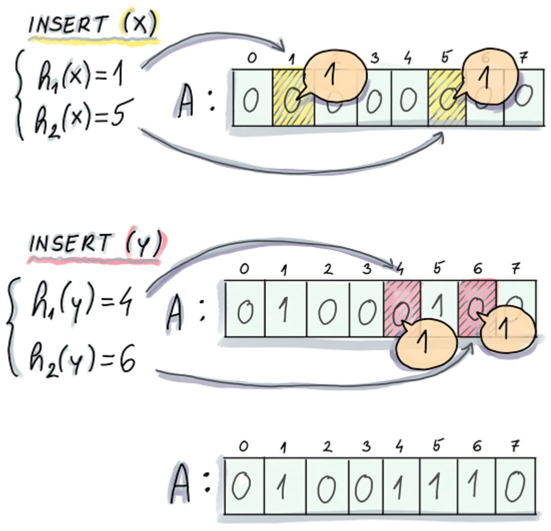

Figure 1: Illustration of a Bloom filter update for $k = 2$. From *Algorithms and Data Structures for Massive Datasets* by Dzejla Medjedovic, Emin Tahirovic, and Ines Dedovic, Manning 2022.

This homework is about a hashing-based data structure called *Bloom filters*, invented by Burton Howard Bloom in 1970.

When we only need to remember whether an element is *present or not* in a set, and do not care for any other information, we can simply set a bit to one in an array of bits. If this array is as large as the universe U of keys, we maintain this information perfectly. Otherwise, we can use a hash function to determine the index of the bit to flip. This leads to a *false positive* problem, where an element seems to be present, while the bit was flipped by another element with the same hash value.

In order to reduce the rate of false positives, we can use k hash functions $h_1, h_2, \ldots, h_k$. When an element $x \in U$ is inserted in the set, we set all the bits of indices $h_1(x), h_2(x), \ldots, h_k(x)$ to one. When we query for the presence of an element, we answer yes whenever all the bits are equal to one. A false positive occurs when we query for an element that is not present, yet all the corresponding bits have been set to one collectively by other inserted elements.

A description and analysis of Bloom filters is given in Chapter 6 of the textbook by Jeff Erickson. It is proved, in particular, that if a table of size m is used to remember n values, then the number k of hash functions that minimizes the false positive rate is (up to integer truncation):

$$k = \frac{m}{n} \cdot \ln 2.$$

Your Work

We ask you to do the following:

- Read Section 6.1 of the textbook Chapter mentioned above.
- Implement a "hash function factory" : a method that takes as input an integer m and returns a hash function that maps integers to $\{0, 1, \ldots, m-1\}$.
- Based on the descriptions above and your hash function factory, implement the Bloom filter data structure. The constructor should involve two parameters: the size m of the array of bits, and the number of hash functions k.
- Experimentally measure the false positive rates for various ranges of the parameters, and validate the above result on the optimal number k of hash functions.

Requirements:

1. A technical report, typeset in LaTeX, with the main points of your implementation, your experimental plan, and your conclusions regarding the outcome of the experiments.
2. The source code of the programs, in an appendix. You have the choice of the programming language, but it makes more sense to use a programming language that allows a fine-grained control on the memory allocation, such as `C++`.

Feel free to play with existing implementations of hash functions, but make sure to explicitly mention any code reuse, and give pointers to the sources.

Deadline

Friday November 7, 2025, before midnight.