Bloom Filters

Bloom Filter, introduced by Bloom in 1970, is a space-efficient randomized data structure for representing a set in order to support membership queries. The set membership problem is as follows. We have a very large $U$, with $|U| = u$. Let $S = \{x_1, x_2, \ldots, x_n\}$ be a subset of $U$ such that $u >> n$. We want a data structure to maintain $S$ that supports membership queries: “Given $x \in U$, is $x \in S$?” The supported operations are

- **INSERT**($x$): $S \leftarrow S \cup \{x\}$
- **QUERY**($x$): is $x \in S$?

**Initial Idea:** We will use a bit vector $B$ of $m = 2n$ bits. Let $h$ be a hash function such that maps each element of $U$ to a random number uniform over the range $\{1, 2, \ldots, m\}$. The above functions are implemented as follows.

\[
\text{INSERT}(x): \quad B[h(x)] \leftarrow 1
\]

\[
\text{QUERY}(x): \quad \text{return } B[h(x)]
\]

Clearly, both operations can be performed in constant time. Note that if $x \in S$ then $\Pr[\text{QUERY}(x)] = 1$ is 1, i.e., our algorithm always gives the correct answer. If $x \notin S$ then we may get a *false positive* as shown below.

$$
\Pr[\text{we output that } x \in S] = \Pr[\text{there is a } y \text{ s.t. } h[y] = h[x]] \leq \sum_{y \in S} \Pr[h(y) = h(x)] = \frac{n}{m} \leq \frac{1}{2}
$$

**Reducing the probability of error.** Suppose we want to reduce the probability of error to $\epsilon$, that is, when $x \notin S$, we want $\Pr[\text{QUERY}(x)] = 1] \leq \epsilon$. To achieve this we will have $k > 1$ tables, each of size $2n$ and each with its own hash function. The two operations now can be implemented as follows.

\[
\text{INSERT}(x):
\begin{align*}
\text{for } i \leftarrow 1 \text{ to } k & \text{ do } \\
B_i[h_i(x)] & \leftarrow 1
\end{align*}
\]

\[
\text{QUERY}(x):
\begin{align*}
\text{for } i \leftarrow 1 \text{ to } k & \text{ do } \\
\text{if } B_i[h_i(x)] = 0 & \text{ then } \\
\text{return 0 } & \\
\text{return 1 }
\end{align*}
\]
Note that if $x \in S$ then $\Pr[\text{QUERY}(x)] = 1$ is 1, i.e., our algorithm always gives the correct answer. If $x \notin S$ then

$$\Pr[\text{we output that } x \in S] = \Pr[\forall 1 \leq i \leq k, \text{there is a } y \text{ s.t. } h_i[y] = h_i[x]]$$

$$\leq \prod_{i=1}^{k} \left[ \sum_{y \in S} \Pr[h_i(y) = h_i(x)] \right]$$

$$= \left( \frac{n}{m} \right)^k$$

$$\leq \left( \frac{1}{2} \right)^k$$

Since we want the error to be at most $\epsilon$, solving for $\left( \frac{1}{2} \right)^k \leq \epsilon$ yields $k \geq \lg(1/\epsilon)$. Thus the time for each operation is $O(\lg(1/\epsilon))$ and the amount of space is $O(n \lg(1/\epsilon))$.

Sometimes Bloom filters are described slightly differently: instead of having different hash tables, there is one array of size $m$ that is shared among all $k$ hash functions.