Review: Encrypted Domain Search
Bloom Filter

Initial with all 0
Bloom Filter

• Assume k hash functions

Each word of document is hashed k times
Each hash location set to 1
Bloom Filter

To check if $y$ is in document, check the $k$ hash locations. If a 0 appears, $y$ is not in document.
Bloom Filter

If only 1s appear, conclude that \( y \) is in \( S \)
This may yield false positive
Parameters & Tradeoffs

• Three parameters
  – Size n/m: bits per keyword
    • n is size of bit vector
    • m is number of distinct keywords to encode
  – k: number of hash functions
    • Affects the computation time
  – Error f: false positive probability

\[ f = (1 - p)^k \approx (1 - e^{-km/n})^k \]
Tradeoffs

• Normally, m is known

• Effect of n
  – Large n: fewer collision; lower false positive

• Effect of k
  – Small k
    • Less computations
    • Actual number of bits (mk) is smaller, so less collision
    • However, fewer bits need to be “collided” to generate a false positive
Bloom Filters and Deletions

- Cannot handle deletions
  - Deleting $x_1$ means resetting 1s to 0s, then deleting $x_1$ will “delete” $x_2$
Counting Bloom Filter

- Track insertions/deletions at hosts
- Send bloom filters (counter may overflow!)

Start with an $m$ bit array, filled with 0s.

Hash each item $x_j$ in $S$ $k$ times. If $H_i(x_j) = a$, add 1 to $B[a]$.

To delete $x_j$ decrement the corresponding counters.

Can obtain a corresponding Bloom filter by reducing to 0/1.