Review: Encrypted Domain Search

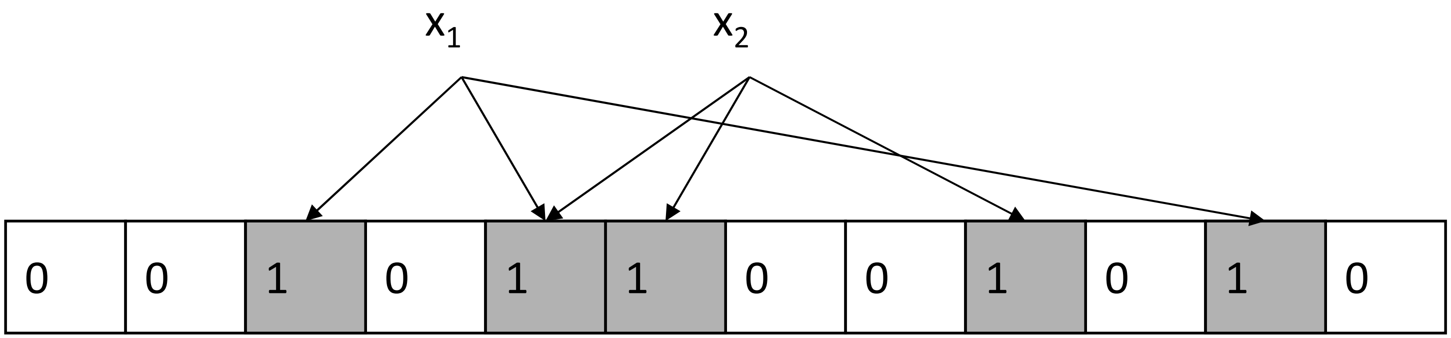
Bloom Filter

0	0	0	0	0	0	0	0	0	0	0	0

Initial with all 0

Bloom Filter

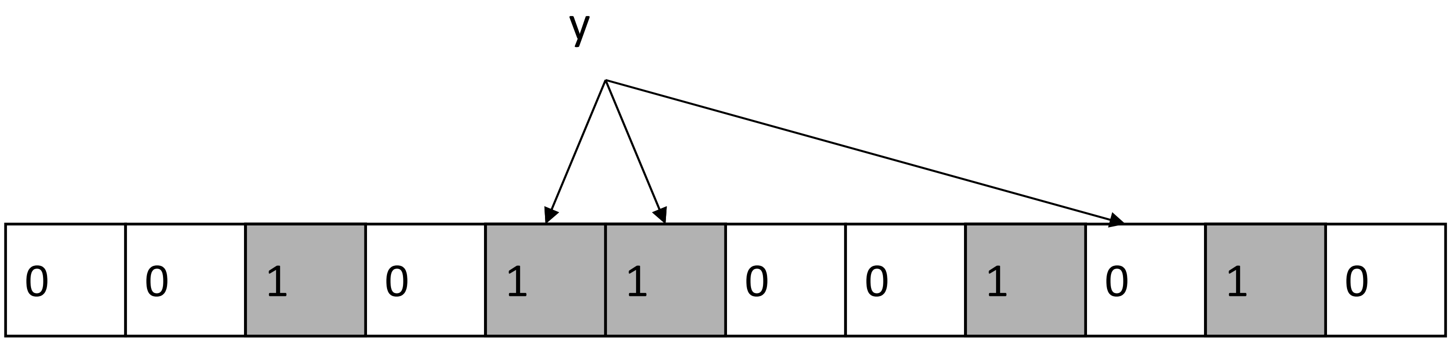
- Assume k hash functions



0	0	1	0	1	1	0	0	1	0	1	0

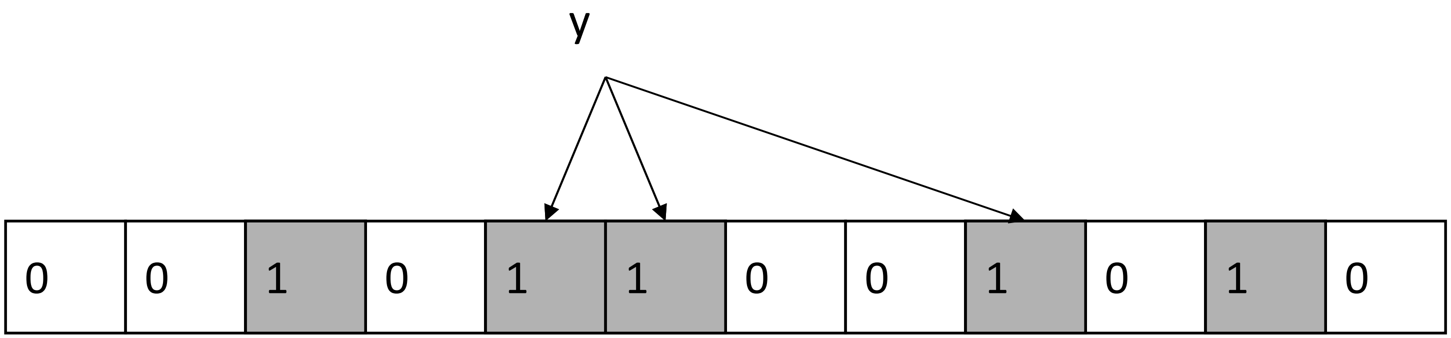
Each word of document is hashed k times
Each hash location set to 1

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



0	0	1	0	1	1	0	0	1	0	1	0

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

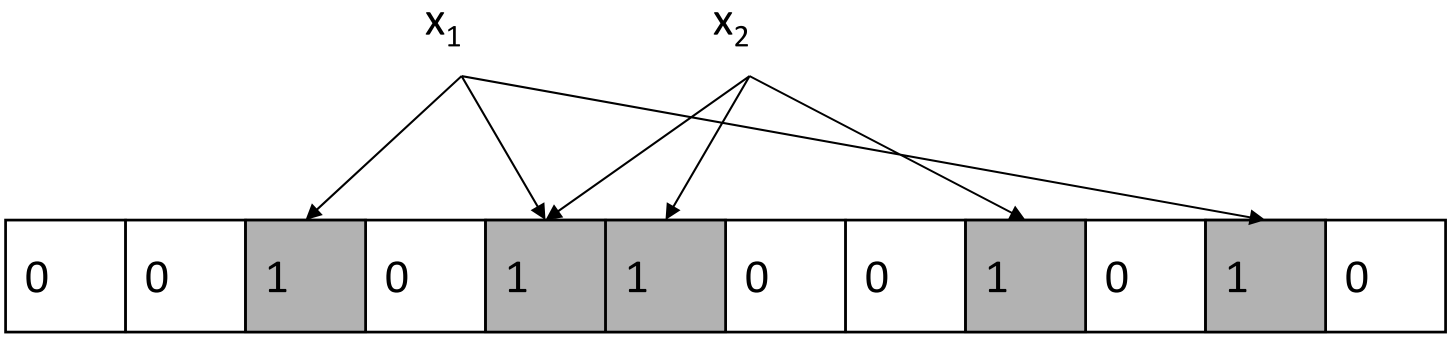
$$f = (1-p)^k \approx (1-e^{-km/n})^k$$

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

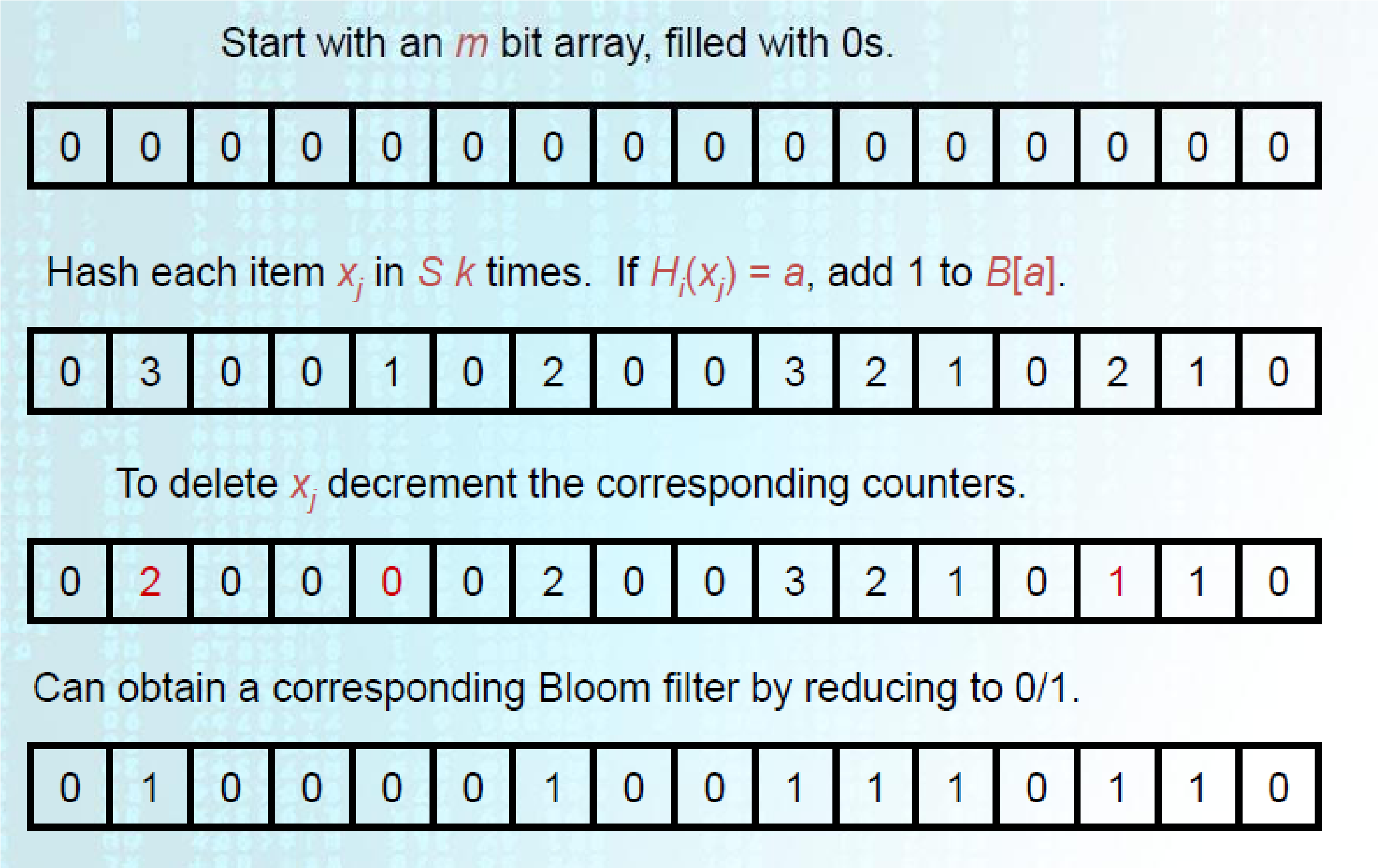
- Cannot handle deletions
  - – Deleting x1 means resetting 1s to 0s, then deleting x1 will "delete" x2



0	0	1	0	1	1	0	0	1	0	1	0

Counting Bloom Filter

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



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

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

0	3	0	0	1	0	2	0	0	3	2	1	0	2	1	0

To delete x_j decrement the corresponding counters.

0	2	0	0	0	0	2	0	0	3	2	1	0	1	1	0

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

0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	0