Bloom Filter

Approximate membership queries

Lookup problem

- Given a set $S = \{x_1, x_2, x_3, \ldots x_n\}$ on a universe U, want to answer queries of the form:

$$\text{is } y \in S\ ?$$

- Example: a set of URLs from the universe of all possible URL strings
- Bloom Filter provides an answer in
  - “Constant” time (time to hash)
  - Small amount of space
  - But with some probability of being wrong

Bloom Filters

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

B	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 $H_i(x_j) = a$, set $B[a] = 1$.

B	0	1	0	0	1	0	1	0	0	1	1	1	0	1	1	0

To check if y is in S, check B at $H_i(y)$. All k values must be 1.

B	0	1	0	0	1	0	1	0	0	1	1	1	0	1	1	0

Possible to have a false positive; all k values are 1, but y is not in S.

B	0	1	0	0	1	0	1	0	0	1	1	1	0	1	1	0

(A toy) Example

- Number of elements n= 2: 9 and 11
- Size of Bloom Filter m=5
- Number of hash functions k=2
  - $h_1(x) = x \bmod 5$
  - $h_2(x) = (2x+3) \bmod 5$

	$h_1(x)$	$h_2(x)$
Initialize		
insert 9	4	1
insert 11	1	0

Bloom Filter

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

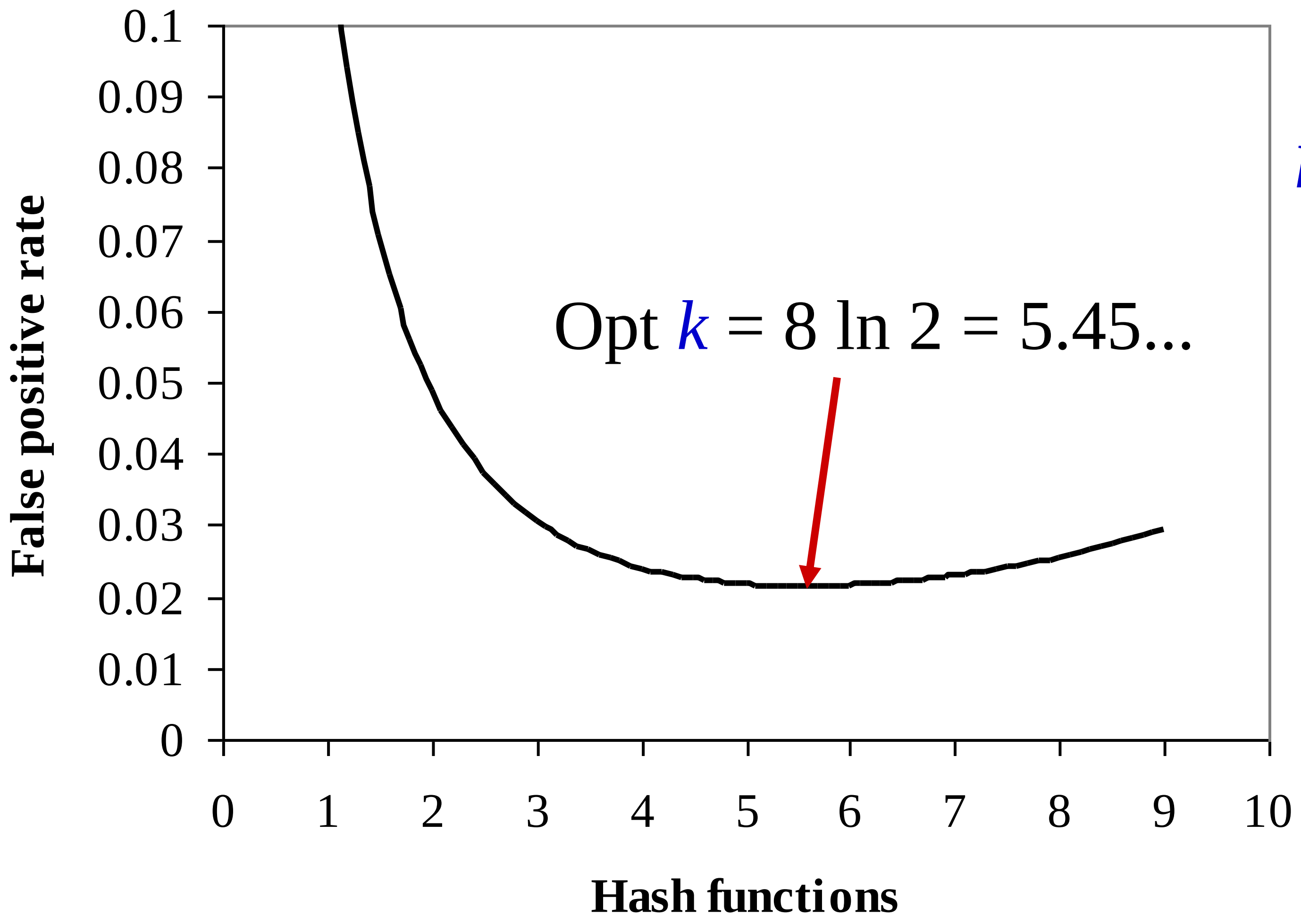
Membership queries

Queries	$h_1(x)$	$h_2(x)$	Answer
for elem 15	0	3	No, not in Bloom Filter (correct answer)
for elem 16	1	0	Yes, in B (wrong answer: false positive)

Errors

- Assumption: We have good hash functions, look random
- Given m bits for filter and n elements, choose number k of hash functions to minimize false positives:
  - Let $p = \Pr[\text{cell is empty}] = (1 - 1/m)^{kn} \approx e^{-kn/m}$
  - Let $f = \Pr[\text{false pos}] = (1 - p)^k \approx (1 - e^{-kn/m})^k$
- As k increases, more chances to find a 0, but more 1's in the array
- Find optimal at $k = (\ln 2)m/n$ by calculus (NOTE prepared by DKatsaros to accompany this lecture)

Example



False positive rate
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
0
1
2
3
4
5
6
7
8
9
10
Hash functions
Opt k = 8 ln 2 = 5.45...

$m/n = 8$