Information Processing Letters 108 (2008) 210–213

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

On the false-positive rate of Bloom filters

Prosenjit Bose *, Hua Guo, Evangelos Kranakis, Anil Maheshwari, Pat Morin, Jason Morrison, Michiel Smid, Yihui Tang

Carleton University, School of Computer Science, 1125 Colonel by Drive, Ottawa, Ontario, Canada

ARTICLE INFO

Article history:
Received 3 May 2004
Available online 26 June 2008
Communicated by F. Dehne

Keywords:
Data structures
Analysis of algorithms

ABSTRACT

Bloom filters are a randomized data structure for membership queries dating back to 1970. Bloom filters sometimes give erroneous answers to queries, called *false positives*. Bloom analyzed the probability of such erroneous answers, called the *false-positive rate*, and Bloom's analysis has appeared in many publications throughout the years. We show that Bloom's analysis is incorrect and give a correct analysis.

1. Introduction

A Bloom filter [2] is an extremely simple randomized data structure for testing membership in a set and has found applications in many areas including (sequential, distributed and parallel) databases [7,12,13], computer networks [3], social networks [9], and cryptography [1,5]. A Bloom filter represents an n element set S using a bit-vector $B = B_1, \ldots, B_m$ of length m. Initially all the bits of B are set to 0. It is assumed that each element x to be stored or searched for comes with a sequence of k random[1] hash values $x_1, \ldots, x_k \in \{1, \ldots, m\}$. To store the element x one simply sets the bits $B_{x_1} = B_{x_2} = \cdots = B_{x_k} = 1$. To query a Bloom filter for an element y one simply checks if $B_{y_1}, \ldots, B_{y_k}$ are all set to 1. If so, the filter outputs "yes" otherwise it outputs "no".

If y is stored in the filter then, by definition, $B_{y_1}, \ldots, B_{y_k}$ are all set to 1, so the query algorithm correctly outputs yes. However, the converse is not true. It is possible that y is not stored in the filter but (by coincidence) $B_{y_1}, \ldots, B_{y_k}$ are all set to 1. This situation is called a *false positive* and the probability that this occurs is called the *false-positive rate*. Bloom derives the false-positive rate in the following way: The probability that any particular bit B_i is equal to 0 is $(1 - 1/m)^{kn}$, since the value i must be avoided by all kn hash values. Therefore, the probability that a particular bit is set to 1 is

$$p \stackrel{\text{def}}{=} \Pr\{B_i = 1\} = 1 - (1 - 1/m)^{kn}.$$

Now, in order for y to result in a false positive, each of the k hash values $y_1, \ldots, y_k$ must be the index of a bit that is set to 1. The probability that this happens is

$$p_{k,n,m} \stackrel{\text{def}}{=} \Pr\{B_{y_1} = 1 \text{ and } B_{y_2} = 1 \text{ and } \ldots \text{and } B_{y_k} = 1\}$$

which is claimed to be

$$p^k = \left(1 - (1 - 1/m)^{kn}\right)^k. \tag{1}$$

This proof, which has appeared in many papers throughout the years, is not quite correct. The error occurs in deriving (1), where there is an implicit assumption that the event "$B_{y_i} = 1$" and the event "$B_{y_1} = B_{y_2} = \cdots = B_{y_{i-1}} = 1$" are independent. At first glance, this seems to be true, since $y_1, \ldots, y_i$ are independent. However, a simple counterexample to this proof can be obtained by considering the case $n = 1$, $k = 2$, $m = 2$. In this case, by simply enumer-

* Corresponding author.
E-mail addresses: jit@scs.carleton.ca (P. Bose), hguo2@scs.carleton.ca (H. Guo), kranakis@scs.carleton.ca (E. Kranakis), maheshwa@scs.carleton.ca (A. Maheshwari), morin@scs.carleton.ca (P. Morin), morrison@scs.carleton.ca (J. Morrison), michiel@scs.carleton.ca (M. Smid), y_tang@scs.carleton.ca (Y. Tang).

[1] Here, as usual, the term "random" means chosen uniformly at random and independently of any other "random" choices.

0020-0190/$ – see front matter
doi:10.1016/j.ipl.2008.05.018

ating the 16 possible situations[2] one finds that the false-positive rate is 5/8, whereas p^k evaluates to $9/16 = 4.5/8$. Thus, Bloom's bound underestimates the false-positive rate in this case.

Mitzenmacher [10] gives a concentration result on the number of 1 bits in Bloom filter that somewhat justifies the standard analysis of Bloom filters for certain common choices of parameters. In this paper, we perform a more detailed analysis of the false-positive rate of Bloom filters. The analysis of the false-positive rate is not nearly as straightforward as one would hope. The contributions of this paper are as follows:

1. We give an exact formula for the false-positive rate of Bloom filters. Unfortunately, this formula is not anywhere near closed form, but could be useful for small values of k, n and m.
2. We give an upper bound on the false-positive rate that converges to p^k for most common choices of the parameters k and m.
3. We show that, rather than being an upper-bound, p^k is actually a strict lower bound on the false-positive rate for any $k \geqslant 2$. That is, for any choice of k, m, and n with $k \geqslant 2$, the false-positive rate of the resulting Bloom filter is greater than p^k.

To the best of our knowledge, this is the first paper to point out this error in the analysis of Bloom filters and give a corrected analysis. The only similar result we know of is a paper by Carter et al. [4] in which they define a data structure (Approximate Membership Tester 1) that, under their assumptions, is equivalent to a Bloom filter. They show that the expected number of bits set to 1 in the filter is mp so the probability of a false positive "is at most about" p^k. Their use of the qualifier "about" indicates that they realize this is not the exact probability, but they do not elaborate any further.

The remainder of this paper is organized as follows: In Section 2 we derive an exact formula for the false-positive rate. In Section 3 we give tight upper and lower bounds on the false-positive rate. In Section 4 we summarize and conclude.

2. An exact formula

We model the problem of determining the false-positive rate as a problem on balls and urns. We are given m urns. We throw kn white balls at random into these urns. We call an urn *white* if it contains at least one white ball. Next we throw k black balls in the urns. Let A be the event that each black ball is in a white urn. We want to evaluate $\Pr\{A\}$. To see that this correctly models Bloom filters, treat the urns as the bits $B_1, \ldots, B_m$ and use the convention that $B_i = 1$ if and only if urn i is white. Thus, the event A corresponds to a false positive (k randomly chosen bits are all set to 1). Thus, the false-positive rate $p_{k,n,m}$ is equal to $\Pr\{A\}$.

[2] There are two elements involved, the element x stored in the table and some element y not stored in the table. Whether or not y is a false positive depends only on x_1, x_2, y_1 and y_2.

Observe that the set of white urns can be represented as a subset of $\{1, \ldots, m\}$. For any $I \subseteq \{1, \ldots, m\}$, let E_I be the event that I is the set of white urns. Observe that $1 \leqslant |I| \leqslant m$. Using conditional probabilities, we get

$$\Pr\{A\} = \sum_{I \subseteq \{1,\ldots,m\}} \Pr\{A \mid E_I\} \cdot \Pr\{E_I\}.$$

If I is fixed then

$$\Pr\{A \mid E_I\} = \left(\frac{|I|}{m}\right)^k,$$

whereas $\Pr\{E_I\}$ is the quotient of

- the number of surjections from a set of size kn onto a set of size i, and
- the number of functions from a set of size kn to a set of size m.

The number of surjections from a set of size kn onto a set of size i is given by $i!\left\{{kn \atop i}\right\}$ where

$$\left\{{kn \atop i}\right\} = \frac{1}{i!}\sum_{j=0}^{i}(-1)^j\binom{i}{j}j^{kn}$$

is called a *Stirling number of the second kind* [6, Section 6.1].

The number of functions from a set of size kn to a set of size m is equal to m^{kn}.

Putting everything together, we obtain

$$\begin{aligned}\Pr\{A\} &= \sum_{I \subseteq \{1,\ldots,m\}} \left(\frac{|I|}{m}\right)^k \times \frac{|I|!\left\{{kn \atop |I|}\right\}}{m^{kn}} \\ &= \frac{1}{m^{k(n+1)}}\sum_{i=1}^{m} i^k i!\binom{m}{i}\left\{{kn \atop i}\right\}.\end{aligned}$$

Theorem 1. *Let $p_{k,n,m}$ be the false-positive rate for a Bloom filter that stores n elements in a bit-vector of size m using k hash functions. Then,*

$$p_{k,n,m} = \frac{1}{m^{k(n+1)}}\sum_{i=1}^{m} i^k i!\binom{m}{i}\left\{{kn \atop i}\right\}.$$

3. Asymptotic bounds

Unfortunately, the formula for $p_{k,n,m}$ given by Theorem 1 is not very enlightening. In particular, it is not easy to compare it directly with p^k, the value derived by Bloom. In this section, we use probability theory to study the asymptotics of $p_{k,n,m}$, and give closed-form upper and lower bounds. We make use of the following result on balls and urns due to Kamath et al. [8] (see also Motwani and Raghavan [11, Theorem 4.18]):

Theorem 2. *(See [8].) Let W denote the number of white urns after throwing kn white balls into m urns. Then*

$$E[W] = m\left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)$$

and for $\lambda > 0$

$$\Pr\{|W - E[W]| \geqslant \lambda\} \leqslant 2\exp\left(-\frac{\lambda^2(m-1/2)}{m^2 - E[W]^2}\right) \leqslant 2\exp\left(-\lambda^2/(2m)\right).$$

Again, let A be the event "every black ball is contained in a white urn". We want to compute upper and lower bounds on $\Pr\{A\}$.

3.1. The upper bound

In this section we give an upper-bound on $p_{k,n,m}$. However, it is awkward (and not very useful) to give an upper bound that holds for all possible choices of k, n and m. Our upper bound requires the condition that

$$\frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}} \leqslant c \tag{2}$$

for some constant $c < 1$. The reasons for this will become apparent in the analysis. To see that this assumption is justified, note that, in nearly all applications of Bloom filters, the parameter k is chosen (as a function of m and n) so that $p = 1-(1-1/m)^{kn}$ is a constant, usually close to $1/2$. Under these conditions, $k = \Theta(m/n)$ and (2) becomes

$$\frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}} = O\left(\frac{m}{n}\sqrt{\frac{m/n + \ln m}{m}}\right) < c.$$

For sufficiently large values of m, this is satisfied as long as $m = o(n^{3/2})$. Again, this is true in all applications of Bloom filters since, if we are willing to use $m = \Theta(n\log n)$ bits of storage, hash tables are a better alternative since they offer constant time searches with no false positives.

We obtain the upper bound by conditioning on the value of W which, according to Theorem 2 is strongly concentrated around its expected value. Recall the definition $p = 1-(1-1/m)^{kn}$ and let $j = E[W] + \sqrt{m(\ln m - 2k\ln p)}$. Then

$$\begin{aligned}
\Pr\{A\} &= \Pr\{W \leqslant j\} \times \Pr\{A \mid W \leqslant j\} \\
&\quad + \Pr\{W > j\} \times \Pr\{A \mid W > j\} && (3)\\
&\leqslant 1 \times \Pr\{A \mid W = j\} + \Pr\{W > j\} \times 1 && (4)\\
&\leqslant \left(\frac{E[W] + \sqrt{m(\ln m - 2k\ln p)}}{m}\right)^k \\
&\quad + 2\exp\left(-\frac{m(\ln m - 2k\ln p)}{2m}\right) && (5)\\
&= \left(p + \sqrt{\frac{\ln m - 2k\ln p}{m}}\right)^k + 2p^k/\sqrt{m} && (6)\\
&\leqslant \sum_{i=0}^{k} p^{k-i}\left(k\sqrt{\frac{\ln m - 2k\ln p}{m}}\right)^i + 2p^k/\sqrt{m} && (7)\\
&\leqslant p^k \times \left(\sum_{i=0}^{k}\left(\frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}}\right)^i + 2/\sqrt{m}\right) && (8)\\
&= p^k \times \left(\frac{1 - \left(\frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}}\right)^{k+1}}{1 - \frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}}} + 2/\sqrt{m}\right) && (9)\\
&\leqslant p^k \times \left(\frac{1}{1 - \frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}}} + 2/\sqrt{m}\right) && (10)\\
&= p^k \times \left(1 + \frac{\frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}}}{1 - \frac{k}{p}\sqrt{\frac{\ln m - 2k\ln p}{m}}} + 2/\sqrt{m}\right) && (11)\\
&= p^k \times \left(1 + O\left(\frac{k}{p}\sqrt{\frac{\ln m - k\ln p}{m}}\right)\right), && (12)
\end{aligned}$$

where (7) uses the inequality (easily verified by induction on k) which states that

$$(a+b)^k \leqslant a^k + kb(a+b)^{k-1}, \quad \text{valid for } a, b \geqslant 0$$

and that, when iterated $k+1$ times gives

$$(a+b)^k \leqslant \sum_{i=0}^{k} a^{k-i}b^i k!/(k-i)! \leqslant \sum_{i=0}^{k} a^{k-i}(bk)^i,$$

valid for $a, b \geqslant 0$.

3.2. The lower bound

For the lower bound, we use a very different argument. Let $b_1, \ldots, b_k$ be the urns in which the k black balls are thrown. We will show that, for $2 \leqslant i \leqslant k$,

$$\begin{aligned}
&\Pr\{b_i \text{ is white} \mid b_1, \ldots, b_{i-1} \text{ are white}\} \\
&\quad > \Pr\{b_i \text{ is white}\} = p. && (13)
\end{aligned}$$

Therefore,

$$\Pr\{A\} = \prod_{i=1}^{k} \Pr\{b_i \text{ is white} \mid b_1, \ldots, b_{i-1} \text{ are white}\} > p^k.$$

Note that this lower-bound is strict, so the actual false-positive rate of a Bloom filter is strictly greater than p^k whenever $k \geqslant 2$. To finish the proof, all that remains is to justify (13). Recall that $b_1, \ldots, b_{i-1}$ are just randomly chosen urns. For any $j \geqslant 2$, the following is obvious

$$\begin{aligned}
&\Pr\{b_1, \ldots, b_{i-1} \text{ are white} \mid W \geqslant j\} \\
&\quad > \Pr\{b_1, \ldots, b_{i-1} \text{ are white}\}. && (14)
\end{aligned}$$

We say that this is obvious because, for example, the case in which all white balls land in one urn is excluded. From the definition of conditional probability, (14) is equivalent to

$$\Pr\{W \geqslant j \mid b_1, \ldots, b_{i-1} \text{ are white}\} > \Pr\{W \geqslant j\}.$$

The above statement says that the random variable W conditioned on "$b_1, \ldots, b_{i-1}$ are white" *stochastically dominates* the random variable W (conditioned on nothing). Note that, if a random variable X stochastically dominates a random variable Y then $E[X] > E[Y]$. Therefore,

$$E[W \mid b_1, \ldots, b_{i-1} \text{ are white}] > E[W].$$

Consider the random variable W/m and observe that $E[W/m] = \Pr\{b_i \text{ is white}\}$. Therefore, we have

$$\begin{aligned}&\Pr\{b_i \text{ is white} \mid b_1, \ldots, b_{i-1} \text{ are white}\}\\ &\quad = E[W/m \mid b_1, \ldots, b_{i-1} \text{ are white}]\\ &\quad > E[W/m]\\ &\quad = \Pr\{b_i \text{ is white}\}\end{aligned}$$

as required for (13).

This completes the proof of

Theorem 3. *Let $p_{k,n,m}$ be the false-positive rate for a Bloom filter that stores n elements in a bit-vector of size m using k hash functions, where $k \geqslant 2$ and k, n and m satisfy* (2)*. Let $p = 1 - (1 - 1/m)^{kn}$. Then,*

$$p^k < p_{k,n,m} \leqslant p^k \times \left(1 + O\left(\frac{k}{p}\sqrt{\frac{\ln m - k \ln p}{m}}\right)\right).$$

4. Conclusions

We have shown that the analysis of Bloom filters originally given by Bloom, and repeated in many subsequent papers, is incorrect. The actual false-positive rate is strictly larger than $p^k = (1 - (1 - 1/m)^{kn})^k$. We have also given bounds on how much larger the false-positive rate can be. Our upper bounds show that, for large enough values of m with small values of k, the difference between p^k and the actual false-positive rate is negligible.

Mullin [12] and Gremillion [7] both observe that the false-positive rate of Bloom filters in their database applications are slightly higher than p^k. However, they attribute this to poor quality pseudorandom numbers. Our results offer another possible explanation: the actual false-positive rate is higher than p^k, even if perfect random numbers are available.

Acknowledgements

The authors are grateful to Michael Mitzenmacher for bringing his paper [10] to our attention.

References

[1] S.M. Bellovin, W.R. Cheswick, Privacy-enhanced searches using encrypted Bloom filters, Draft, 2004, http://www.research.att.com/~smb/papers/bloom-encrypt.ps.
[2] B.H. Bloom, Space/time trade-offs in hash coding with allowable errors, Communications of the ACM 13 (7) (1970) 422–426.
[3] A. Broder, M. Mitzenmacher, Network applications of Bloom filters: A survey, in: Proceedings of the 40th Annual Allerton Conference on Communication, Control and Computing, 2002, pp. 636–646.
[4] L. Carter, R. Floyd, J. Gill, G. Markowsky, M. Wegman, Exact and approximate membership testers, in: Annual ACM Symposium on Theory of Computing, 1978, pp. 59–65.
[5] E.-J. Goh, Secure indexes for efficient searching on encrypted compressed data, Technical Report 2003/216, Cryptology ePrint Archive, 2003. http://eprint.iacr.org/2003/216/.
[6] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics, second ed., Addison-Wesley, 1994.
[7] L.L. Gremillion, Designing a Bloom filter for differential access, Communications of the ACM 25 (7) (1982) 600–604.
[8] A. Kamath, R. Motwani, K. Palem, P. Spirakis, Tail bounds for occupancy and the satisfiability threshold conjecture, in: Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science, 1994, pp. 592–603.
[9] J. Schachter M. Ceglowski, Loaf, Online at http://loaf.cantbedone.org/.
[10] M. Mitzenmacher, Compressed Bloom filters, IEEE/ACM Transactions on Networks 10 (5) (2002).
[11] R. Motwani, P. Raghavan, Randomized Algorithms, Cambridge University Press, 1995.
[12] J.K. Mullin, A second look at Bloom filters, Communications of the ACM 26 (8) (1983).
[13] M.V. Ramakrishna, Practical performance of Bloom filters and parallel free-text searching, Communications of the ACM 25 (7) (1982) 600–604.