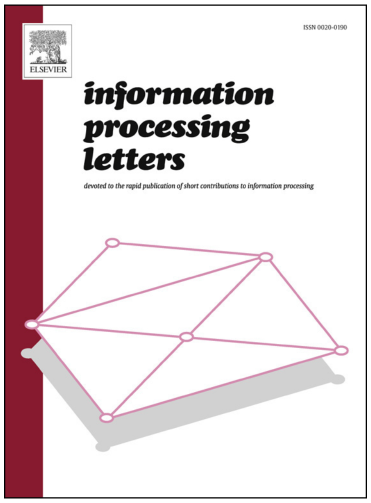
On the analysis of Bloom filters

Fabio Grandi



PII: S0020-0190(17)30160-6
DOI: https://doi.org/10.1016/j.ipl.2017.09.004
Reference: IPL 5580

To appear in: *Information Processing Letters*

Received date: 8 August 2017
Accepted date: 14 September 2017

Highlights

- Probabilistic analysis of Bloom filters is completed and extended.
- Two kinds of Bloom filter, standard and classic, are considered.
- Iterative schemes for computing the false positive probability are provided.
- A new accurate approximation for the false positive probability is provided.

On the analysis of Bloom filters

Fabio Grandi[a,*]

[a]*Department of Computer Science and Engineering (DISI), Alma Mater Studiorum – Università di Bologna, Viale Risorgimento 2, I-40136 Bologna BO, Italy*

Abstract

The Bloom filter is a simple random binary data structure which can be efficiently used for approximate set membership testing. When testing for membership of an object, the Bloom filter may give a false positive, whose probability is the main performance figure of the structure. We complete and extend the analysis of the Bloom filter available in the literature by means of the γ-*transform* approach. Known results are confirmed and new results are provided, including the variance of the number of bits set to 1 in the filter. We consider the choice of bits to be set to 1 when an object is inserted both with and without replacement, in what we call *standard* and *classic* Bloom filter, respectively. Simple iterative schemes for the computation of the false positive probability and a new non-iterative approximation, taking into account the variance of bits set to 1, are also provided.

Keywords: Data structures, Analysis of algorithms, Bloom filters, γ-transform

1. Introduction

The Bloom filter [1] is a simple random data structure which can be efficiently used for approximate set membership testing. Considering n objects $o_i \in O$ ($i \in \{1..n\}$) to be inserted in a Bloom filter made of m bits initially set to 0, k independent hash functions $h_j : O \to \{1..m\}$ ($j \in \{1..k\}$) are used to map each object into bit positions to be set to 1 in the filter. In order to test the membership of an object $o \in O$ to the set $\{o_1, \ldots, o_n\}$, the k hash functions can be applied to o: in case at least one maps o to the position of a bit still 0 in the filter, then the membership can be excluded. If o is mapped to bits all set to 1, then o can be one of the objects in the set but we can also be in the presence of a *false positive*. A low False Positive Probability (FPP) is, thus, a quality figure of the filter that has to be minimized via a suitable choice and tuning of the parameters (m, n, k).

In the *standard* Bloom filter usually considered in the recent literature and in the application practice, there are no constraints imposed to the values gen-

*Corresponding author
Email address: `fabio.grandi@unibo.it` (Fabio Grandi)

erated by the k hash functions, so that the same values can be repeatedly generated and less than k bits can be set by the insertion of an object in the filter. In this work, we also consider the variant initially proposed by Bloom [1] in which, for each object, the k hash functions always generate k distinct values and, thus, exactly k bits are set in the filter, as required for the classic superimposed coding [9]. Hence, we will call such variant the *classic* Bloom filter (if the hash functions have disjoint ranges of m/k consecutive bits, this variant corresponds to what has been called *partitioned* Bloom filter in [7]). Notice that the adoption of a classic Bloom filter does not give rise to significant additional computational costs, with respect to a standard Bloom filter, by exploiting the techniques introduced in [7] to avoid hash collisions.

1.1. Background on Approximate Analysis

After all objects have been inserted, the probability that one bit of the standard Bloom filter is still 0 can be evaluated as $(1 - 1/m)^{kn}$, being the selection of bits to be set with replacement, either with respect to the objects and with respect to the hash functions. If X is a r.v. representing the total number of bits set to 1 in the filter, its expected value is accordingly:

$$\mathrm{E}[X] \quad = \quad m\left[1-\left(1-\frac{1}{m}\right)^{kn}\right] \tag{1}$$

The main merit figure of the Bloom filter is the False Positive Probability (FPP) that can be computed as the probability that an object non belonging to $\{o_1, \ldots, o_n\}$ is hashed to only positions with bits set to 1 in the filter. As the bit positions can be chosen with replacement, the probability of a false positive conditioned to a number $X = x$ of bits set to 1 in the standard Bloom filter is given by:

$$\Pr(\mathrm{FP}|X=x) \quad = \quad \left(\frac{x}{m}\right)^{k} \tag{2}$$

Then the exact value of the FPP can be computed indeed according to the Total Probability theorem as:

$$\mathrm{FPP} \quad = \quad \sum_{x=0}^{m} \Pr(\mathrm{FP}|X=x)\Pr(X=x) \quad = \quad \sum_{x=0}^{m} f(x)\Pr(\mathrm{FP}|X=x) \tag{3}$$

where $f(x)$ is the probability mass function of X. Since X can be shown (e.g., via application of the Azuma–Hoeffding inequality [8, Sec. 12.5.3]) to be strongly concentrated around its expected value, a commonly employed approximation is to consider x deterministically equal to $\mathrm{E}[X]$, yielding:

$$\mathrm{FPP} \quad \approx \quad \mathrm{FPP}_{A1} \quad = \quad \left[1-\left(1-\frac{1}{m}\right)^{kn}\right]^{k} \tag{4}$$

Such approximation has been shown in [2] to be highly accurate for large m values with small values of k. Moreover, since $(1-1/m)^m \to 1/e$ when m grows, a further asymptotic approximation:

$$\text{FPP} \approx \text{FPP}_{A2} = \left(1 - e^{-kn/m}\right)^k \tag{5}$$

is also commonly used when m is large. FPP_{A2} is minimized when $k = (m/n)\ln 2$, corresponding to one half of the bits set to 1 in the Bloom filter.

No complete analysis of the classic (or partitioned) Bloom filter has been done yet. Kirsch and Mitzenmacher in [7] limit themselves to observe that it tends to have more 1's than the standard Bloom filter and, thus, yields an higher FPP although their asymptotic behavior is the same.

In this paper, we will apply the γ-transform approach described in [5] and that we first introduced in [4] to the analysis both of the standard and of the classic Bloom filters. In this way, in Sec. 2, we will easily derive the exact probability mass function, expected value and variance of the number of bits set to 1, and the FPP of the standard and classic Bloom filters. We will also introduce two iterative schemes for the direct computation of those FPPs and a new accurate non-iterative approximation for the estimation of the FPP of the standard Bloom filter. For small Bloom filters, for which asymptotic approximations are not justified, a comparison between the FPPs of the standard and classic Bloom filters, the new approximation and the old ones can be found in Sec. 3. A Conclusion section will finally close the paper.

2. A New Analysis of Bloom Filters

In this Section, we exploit the γ-transform approach [4, 5] for the probabilistic characterization of the standard and classic Bloom filters. In a counting experiment where possible outcomes can be selected from a set with cardinality m, the γ-transform $\gamma(y)$ of the probability mass function of the number of outcomes can be evaluated as the probability of selecting outcomes from a subset with cardinality $y \le m$ only. In our case, we can consider as an outcome a bit set to 1 in the filter so that X represents the number of outcomes. Ready-made formulas will then allow us to derive from $\gamma(y)$ the probability mass function of X, the expected value and variance of X.

2.1. Standard Bloom Filter

Owing to the physical meaning of the γ-transform recalled above [5, Th. 3], since in the standard Bloom filter selection of bits to be set to 1 is with replacement, we have $\gamma_S(y) = (y/m)^{kn}$. Hence, using formulae (6), (13) and (14) of [5], we can derive in a straightforward way from $\gamma_S(y)$ the probability mass function, expected value and variance of X, respectively, as:

$$f_S(x) = \binom{m}{x} \sum_{j=0}^{x} (-1)^j \binom{x}{j} \left(\frac{x-j}{m}\right)^{kn} \tag{6}$$

$$\mathrm{E}[X] = m\left[1-\left(1-\frac{1}{m}\right)^{kn}\right] \quad (7)$$

$$\sigma_X^2 = m\left(1-\frac{1}{m}\right)^{kn}\left[1-m\left(1-\frac{1}{m}\right)^{kn}+(m-1)\left(1-\frac{1}{m-1}\right)^{kn}\right] \quad (8)$$

The probability mass function $f_S(x)$ is the one we first derived in [4] for a particular case of the "set union problem" and agrees with the expressions derived for the standard Bloom filter in [2, 3], while $\mathrm{E}[X]$ in (7) is the same as in (1). As far as we know, no derivation of σ_X^2 has been done by other authors. Notice that the explicit knowledge of σ_X^2 is a good indicator for evaluating how the distribution of X is actually concentrated around $\mathrm{E}[X]$ and, thus, of the goodness of the proposed approximations FPP_{A1} and FPP_{A2} (also for small m).

Using (3) with (6) and (2), the exact expression of the FPP for the standard Bloom filter can then be computed as:

$$\mathrm{FPP}_S = \sum_{x=0}^{m}\left(\frac{x}{m}\right)^k\binom{m}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m}\right)^{kn} \quad (9)$$

which agrees with the expressions derived in [2, 3] and is a rather complex formula to evaluate.

2.1.1. Iterative Computation of FPP_S

Whereas in [3] an iterative scheme has been provided for the computation of $f_S(x)$, we present a similar iterative scheme for direct computation of FPP_S. Let $\psi_S(h,m)$ be a bivariate function with integer arguments defined as:

$$\psi_S(h,m) = \sum_{x=0}^{m}\left(\frac{x}{m}\right)^h\binom{m}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m}\right)^{kn}$$

Then we can show that $\psi_S(h,m)$ satisfies the difference equation that follows:

$$\psi_S(h,m) = \psi_S(h-1,m)-\left(1-\frac{1}{m}\right)^{kn+h-1}\psi_S(h-1,m-1) \quad (10)$$

Proof: Since

$$\left(\frac{x}{m}\right)^h\binom{m}{x} = \left(\frac{x}{m}\right)^h\frac{m}{x}\binom{m-1}{x-1} = \left(\frac{x}{m}\right)^{h-1}\left[\binom{m}{x}-\binom{m-1}{x}\right]$$

we can write:

$$\psi_S(h,m) = \sum_{x=0}^{m}\left(\frac{x}{m}\right)^{h-1}\binom{m}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m}\right)^{kn} - \sum_{x=0}^{m-1}\left(\frac{x}{m}\right)^{h-1}\binom{m-1}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m}\right)^{kn}$$

$$\begin{aligned}
&= \sum_{x=0}^{m}\left(\frac{x}{m}\right)^{h-1}\binom{m}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m}\right)^{kn} \\
&\quad -\sum_{x=0}^{m-1}\left(\frac{m-1}{m}\right)^{h-1}\left(\frac{x}{m-1}\right)^{h-1}\binom{m-1}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{m-1}{m}\right)^{kn}\left(\frac{x-j}{m-1}\right)^{kn} \\
&= \sum_{x=0}^{m}\left(\frac{x}{m}\right)^{h-1}\binom{m}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m}\right)^{kn} \\
&\quad -\left(1-\frac{1}{m}\right)^{kn+h-1}\sum_{x=0}^{m-1}\left(\frac{x}{m-1}\right)^{h-1}\binom{m-1}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left(\frac{x-j}{m-1}\right)^{kn} \\
&= \psi_S(h-1,m)-\left(1-\frac{1}{m}\right)^{kn+h-1}\psi_S(h-1,m-1)
\end{aligned}$$

which concludes the proof. Hence, since $\text{FPP}_S = \psi_S(k,m)$, the difference scheme (10) with the initial conditions $\psi_S(0,i) = 1$ $(i \in \{1..m\})$ gives us an $O(km)$ iterative algorithm for computing FPP_S.

2.1.2. *A new non-iterative approximation of* FPP_S

A non-iterative better approximation than (4) can also be determined as follows. Being X the r.v. representing the number of bits set to 1, from (2) and (3), FPP_S is the expected value of a function of X defined as $\varphi(X) = (X/m)^k$. Instead of simply evaluating FPP_S as $\varphi(\text{E}[X])$ as in (4), which is a zero-th order approximation, we can develop φ in Taylor's series around $\text{E}[X]$, keeping the first three terms, and take the expectation yielding:

$$\begin{aligned}
\text{E}[\varphi(X)] &\approx \varphi(\text{E}[X]) + \frac{\sigma_X^2}{2}\,\varphi''(\text{E}[X]) \\
&= \left(\frac{\text{E}[X]}{m}\right)^k + \frac{\sigma_X^2}{2}\,\frac{k(k-1)}{m^2}\left(\frac{\text{E}[X]}{m}\right)^{k-2} \qquad (11)
\end{aligned}$$

which is a second order approximation, which we will call FPP_{A3}, also taking into account the non-null variance of X and which can be computed using the values of $\text{E}[X]$ and σ_X^2 given by (7) and (8), respectively. Notice that, although the expected value (7) can easily be computed without resorting to the γ-transform approach, the direct computation of the variance (8) from the probability mass function (6) is a rather hard task.

2.2. *Classic Bloom Filter*

In this section we analyze a Bloom filter as originally proposed in [1], that is where the insertion of each object sets exactly k bits to 1. Hence, the selection of bits to be set to 1 is with replacement with respect to objects but without replacement with respect to hash functions. Therefore, owing to the physical meaning of the γ-transform [5, Th. 3], we can derive $\gamma_C(y) = \left[\binom{y}{k}/\binom{m}{k}\right]^n$.

Hence, using formulae (6), (13) and (14) of [5], we can derive from $\gamma_C(y)$ the probability mass function, expected value and variance of X, respectively, as:

$$f_C(x) = \binom{m}{x}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left[\frac{\binom{x-j}{k}}{\binom{m}{k}}\right]^n \tag{12}$$

$$\mathrm{E}[X] = m\left[1-\left(1-\frac{k}{m}\right)^n\right] \tag{13}$$

$$\sigma_X^2 = m\left(1-\frac{k}{m}\right)^n\left[1-m\left(1-\frac{k}{m}\right)^n+(m-1)\left(1-\frac{k}{m-1}\right)^n\right] \tag{14}$$

As far as we know, no complete characterization of the classic Bloom filter has been done before, although the probability mass function $f_C(x)$ and the expected value (13) are special cases of those presented in [9, 5]. The variance (14) agrees with the one we derived in [4, 5] for the general "set union problem".

The exact false positive probability of the classic Bloom filter can be computed as the probability that an object non belonging to $\{o_1,\ldots,o_n\}$ is hashed to only positions with bits set to 1 in the filter when the selection of such positions is without replacement. Hence, from (3) and using (12), we have:

$$\mathrm{FPP}_C = \sum_{x=0}^{m} f_C(x)\frac{\binom{x}{k}}{\binom{m}{k}} \tag{15}$$

$$= \sum_{x=0}^{m}\binom{m}{x}\frac{\binom{x}{k}}{\binom{m}{k}}\sum_{j=0}^{x}(-1)^j\binom{x}{j}\left[\frac{\binom{x-j}{k}}{\binom{m}{k}}\right]^n \tag{16}$$

Notice that (15) can also be rewritten as:

$$\mathrm{FPP}_C = \sum_{x=0}^{m} f_C(x)\frac{x^{\underline{k}}}{m^{\underline{k}}} = \frac{1}{m^{\underline{k}}}\,\mathrm{E}[X^{\underline{k}}] \tag{17}$$

where $a^{\underline{k}}$ is the k-th falling factorial power of a and $\mathrm{E}[X^{\underline{k}}]$ is the k-th factorial moment of X, which can be easily evaluated via the γ-transform approach. In fact, using formula (12) of [5] we obtain:

$$\mathrm{FPP}_C = \sum_{j=0}^{k}(-1)^j\binom{k}{j}\gamma_C(m-j) = \sum_{j=0}^{k}(-1)^j\binom{k}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n \tag{18}$$

which is a more handy formulation than (16), still complex to evaluate though.

2.2.1. Iterative Computation of FPP$_C$

As we did for FPP$_S$, we also present here an iterative scheme for direct computation of FPP$_C$. Let $\psi_C(h,m)$ be a bivariate function with integer arguments defined as:

$$\psi_C(h,m) \;=\; \sum_{j=0}^{h}(-1)^j\binom{h}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n$$

Then we can show that $\psi_C(h,m)$ satisfies the difference equation that follows:

$$\psi_C(h,m) \;=\; \psi_C(h-1,m)-\left(1-\frac{k}{m}\right)^n\psi_C(h-1,m-1) \qquad (19)$$

Proof:

$$\begin{aligned}
\psi_C(h,m) &= 1+\sum_{j=1}^{h}(-1)^j\binom{h}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n \\
&= 1+\sum_{j=1}^{h}(-1)^j\binom{h-1}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n+\sum_{j=1}^{h}(-1)^j\binom{h-1}{j-1}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n \\
&= 1+\sum_{j=1}^{h-1}(-1)^j\binom{h-1}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n+\sum_{j=1}^{h}(-1)^j\binom{h-1}{j-1}\left[\frac{\binom{m-1-(j-1)}{k}}{\frac{m}{m-k}\binom{m-1}{k}}\right]^n \\
&= \sum_{j=0}^{h-1}(-1)^j\binom{h-1}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n+\left(\frac{m-k}{m}\right)^n\sum_{j=1}^{h}(-1)^j\binom{h-1}{j-1}\left[\frac{\binom{m-1-(j-1)}{k}}{\binom{m-1}{k}}\right]^n \\
&= \sum_{j=0}^{h-1}(-1)^j\binom{h-1}{j}\left[\frac{\binom{m-j}{k}}{\binom{m}{k}}\right]^n-\left(\frac{m-k}{m}\right)^n\sum_{j=0}^{h-1}(-1)^j\binom{h-1}{j}\left[\frac{\binom{m-1-j}{k}}{\binom{m-1}{k}}\right]^n \\
&= \psi_C(h-1,m)-\left(1-\frac{k}{m}\right)^n\psi_C(h-1,m-1)
\end{aligned}$$

which concludes the proof. Hence, since FPP$_C$ $=\psi_C(k,m)$, the difference scheme (19) with the initial conditions $\psi_C(0,i)=1$ ($i\in\{1..m\}$) gives us an $O(km)$ iterative algorithm for computing FPP$_C$. However, if $k\ll m$, it is more convenient to directly evaluate formula (18) which has complexity $O(k^2)$.

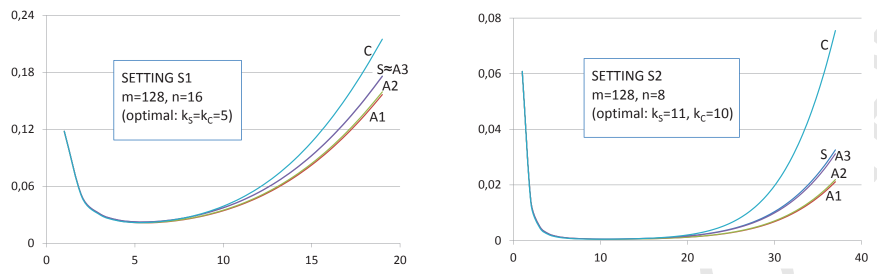


Figure 1: Graphs of False Positive Probabilities and their approximations.

3. Comparison

In this Section we analyze —away from their asymptotic behavior, that is for rather small values of m— the exact FPPs and the approximations we presented for the Bloom filters. Notice that, as also underlined in [3], although large Bloom filters are usually adopted, there are also applications actually using small Bloom filters (e.g., [6]). We considered several (m, n) combinations with $m \leq 1024$, for which we obtained qualitatively similar results. We present in detail the analysis of two representative settings with $m = 128$: S1 with $n = 16$ and S2 with $n = 8$. Fig. 1 displays the plot versus k of the FPPs and approximations in the range around their minima, which is the one of interest for applications, for these two settings. In particular, from the top to the bottom, the plotted curves are: FPP_C, FPP_S and its approximations FPP_{A3}, FPP_{A2} and FPP_{A1}.

Approximations FPP_{A1} and FPP_{A2} of FPP_S are very close to each other on the whole range of k (their maximal error is, resp., 11.18% and 9.50% occurring at $k = 19$ in S1 and, resp., 35.33% and 32.84% occurring at $k = 35$ in S2; in the optimal configuration their error is, resp., 3.82% and 2.45% in S1 and 14.52% and 11.92% in S2). The new approximation FPP_{A3} is better than FPP_{A1} and FPP_{A2} and very close to FPP_S indeed (its maximal error is 0.28% occurring at $k = 39$ in S1 and 4,53% occurring at $k = 33$ in S2; in the optimal configuration its error is 0.0044% in S1 and and 0.37% in S2). In S1, the optimal configuration corresponds to $k = 5$ for both the standard and the classic Bloom filters (minimum FPP_S and minimum FPP_C are 0.023 and 0.022, resp., with a 3.28% difference). In S2, the optimal configuration corresponds to $k = 11$ for the standard Bloom filter and to $k = 10$ for the classic Bloom filter (minimum FPP_S and minimum FPP_C are 0.00054 and 0.00046, resp., with a 13.59% difference). Hence, in their optimal configuration, the performance difference between the standard and the classic Bloom filters is not so significant. The difference between FPP_C and FPP_S becomes really significant for higher k values (maximum percentage difference between FPP_C and FPP_S is 25.62% at $k = 25$ in S1 and 151.28% at $k = 45$ in S2), usually out of the range of practical application of Bloom filters. In any case, in the optimality region, $\mathrm{FPP}_C, \mathrm{FPP}_S$

and also FPP_{A3} are very close to each other, and they are also quite flat around their minima.

4. Conclusion

In this paper, we applied the γ-transform approach to the probabilistic characterization of the standard and classic Bloom filters. In this way, we easily re-obtained known results (i.e, $f_S(x)$, $\mathrm{E}[X]$ for the the standard and classic Bloom filters, FPP_S) and derived new results (i.e., $f_C(x)$, σ_X^2 for the the standard and classic Bloom filters, FPP_C). We also presented iterative schemes for the direct computation of FPP_S and FPP_C and a new non-iterative approximation FPP_{A3} for FPP_S. For small Bloom filters, it has been shown that FPP_{A3} is a quite accurate approximation, better than FPP_{A1} and FPP_{A2}, and that the performances of the standard and classic Bloom filters, with optimal k, are comparable.

[1] B. H. Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors," *Communications of the ACM*, Vol. 13, No. 7, 1970, pp. 422-426.

[2] P. Bose, H. Guo, E. Kranakis, A. Maheshwari, P. Morin, J. Morrison, M. H. M. Smid, Y. Tang, "On the false-positive rate of Bloom filters," *Information Processing Letters*, Vol. 108, No. 4, 2008, pp. 210–213.

[3] K. J. Christensen, A. Roginsky, M. Jimeno, "A new analysis of the false positive rate of a Bloom filter," *Information Processing Letters*, Vol. 110, No. 21, 2010, pp. 944–949.

[4] F. Grandi, "*Advanced access cost models for databases*," Ph.D. Dissertation, DEIS, University of Bologna, Italy, 1994, *in Italian*.

[5] F. Grandi, "The γ-transform Approach: a New Method for the Study of a Discrete and Finite Random Variable," *International Journal of Mathematical Models and Methods in Applied Sciences*, Vol. 9, 2015, pp. 624–635, URL: `http://www.naun.org/main/NAUN/ijmmas/2015/b442001-411.pdf`.

[6] J. Mullin, "Accessing textual documents using compressed indexes of arrays of small Bloom filters, *The Computer Journal*, Vol. 30, No. 4, 1987, pp. 343–348.

[7] A. Kirsch, M. Mitzenmacher, Less hashing, same performance: Building a better Bloom filter, *Random Structures and Algorithms*, Vol. 33, No. 2, 2008, pp. 187–218.

[8] M. Mitzenmacher, E. Upfal, *Probability and computing: Randomized algorithms and probabilistic analysis*, Cambridge University Press, Cambridge, UK, 2005.

[9] C. S. Roberts, "Partial match retrieval via the method of superimposed codes," *Proceedings of the IEEE*, Vol. 67, No. 12, 1979, pp. 1624–1642.