2007 Conference on Analysis of Algorithms, AofA 07 DMTCS proc. **AH**, 2007, 127–146

HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm

Philippe Flajolet[1] and Éric Fusy[1] and Olivier Gandouet[2] and Frédéric Meunier[1]

[1]*Algorithms Project, INRIA–Rocquencourt, F78153 Le Chesnay (France)*
[2]*LIRMM, 161 rue Ada, 34392 Montpellier (France)*

This extended abstract describes and analyses a near-optimal probabilistic algorithm, HYPERLOGLOG, dedicated to estimating the number of *distinct* elements (the *cardinality*) of very large data ensembles. Using an auxiliary memory of m units (typically, "short bytes"), HYPERLOGLOG performs a single pass over the data and produces an estimate of the cardinality such that the relative accuracy (the *standard error*) is typically about $1.04/\sqrt{m}$. This improves on the best previously known cardinality estimator, LOGLOG, whose accuracy can be matched by consuming only 64% of the original memory. For instance, the new algorithm makes it possible to estimate cardinalities well beyond 10^9 with a typical accuracy of 2% while using a memory of only 1.5 kilobytes. The algorithm parallelizes optimally and adapts to the sliding window model.

Introduction

The purpose of this note is to present and analyse an efficient algorithm for estimating the *number of distinct elements*, known as the *cardinality*, of large data ensembles, which are referred to here as *multisets* and are usually massive *streams* (read-once sequences). This problem has received a great deal of attention over the past two decades, finding an ever growing number of applications in networking and traffic monitoring, such as the detection of worm propagation, of network attacks (e.g., by Denial of Service), and of link-based spam on the web [3]. For instance, a data stream over a network consists of a sequence of packets, each packet having a header, which contains a pair (source–destination) of addresses, followed by a body of specific data; the number of distinct header pairs (the cardinality of the multiset) in various time slices is an important indication for detecting attacks and monitoring traffic, as it records the number of distinct active flows. Indeed, worms and viruses typically propagate by opening a large number of *different* connections, and though they may well pass unnoticed amongst a huge traffic, their activity becomes exposed once cardinalities are measured (see the lucid exposition by Estan and Varghese in [11]). Other applications of cardinality estimators include data mining of massive data sets of sorts—natural language texts [4, 5], biological data [17, 18], very large structured databases, or the internet graph, where the authors of [22] report computational gains by a factor of 500^+ attained by probabilistic cardinality estimators.

Clearly, the cardinality of a multiset can be exactly determined with a storage complexity essentially proportional to its number of elements. However, in most applications, the multiset to be treated is far too large to be kept in core memory. A crucial idea is then to relax the constraint of computing the value n of the cardinality *exactly*, and to develop probabilistic algorithms dedicated to *estimating* n *approximately*. (In many practical applications, a tolerance of a few percents on the result is acceptable.) A whole range of algorithms have been developed that only require a sublinear memory [2, 6, 10, 11, 15, 16], or, at worst a linear memory, but with a small implied constant [24].

All known efficient cardinality estimators rely on randomization, which is ensured by the use of *hash* functions. The elements to be counted belonging to a certain data domain $\mathcal{D}$, we assume given a *hash function*, $h : \mathcal{D} \to \{0, 1\}^\infty$; that is, we assimilate hashed values to infinite binary strings of $\{0, 1\}^\infty$, or equivalently to real numbers of the unit interval. (In practice, hashing on 32 bits will suffice to estimate cardinalities in excess of 10^9; see Section 4 for a discussion.) We postulate that the hash function has been designed in such a way that the hashed values closely resemble a uniform model of randomness, namely, *bits of hashed values are assumed to be independent and to have each probability* $\frac{1}{2}$ *of occurring*— practical methods are known [20], which vindicate this assumption, based on cyclic redundancy codes (CRC), modular arithmetics, or a simplified cryptographic use of boolean algebra (e.g., `sha1`).

The best known cardinality estimators rely on making suitable, concise enough, observations on the hashed values $h(\mathcal{M})$ of the input multiset $\mathcal{M}$, then inferring a plausible estimate of the unknown cardinality n. Define an *observable* of a multiset $S \equiv h(\mathcal{M})$ of $\{0, 1\}^\infty$ strings (or, equivalently, of real $[0, 1]$ numbers) to be a function that only depends on the *set* underlying S, that is, a quantity independent of replications. Then two broad categories of cardinality observables have been studied.

— *Bit-pattern observables*: these are based on certain patterns of bits occurring at the beginning of the (binary) S-values. For instance, observing in the stream S at the beginning of a string a bit-pattern $0^{\rho-1}1$ is more or less a likely indication that the cardinality n of S is at least 2^ρ. The algorithms known as *Probabilistic Counting*, due to Flajolet-Martin [15], together with the more recent LOGLOG of Durand-Flajolet [10] belong to this category.

— *Order statistics observables*: these are based on order statistics, like the smallest (real) values, that appear in S. For instance, if $X = \min(S)$, we may legitimately hope that n is roughly of the order of $1/X$, since, as regards expectations, one has $\mathbb{E}(X) = 1/(n + 1)$. The algorithms of Bar-Yossef *et al.* [2] and Giroire's MINCOUNT [16, 18] are of this type.

The observables just described can be maintained with just one or a few registers. However, as such, they only provide a rough indication of the sought cardinality n, via $\log_2 n$ or $1/n$. One difficulty is due to a rather high variability, so that one observation, corresponding to the maintenance of a *single* variable, cannot suffice to obtain accurate predictions. An immediate idea is then to perform several experiments in parallel: if each of a collection of m random variables has standard deviation σ, then their arithmetic mean has standard deviation $\sigma/\sqrt{m}$, which can be made as small as we please by increasing m. That simplistic strategy has however two major drawbacks: it is costly in terms of computation time (we would need to compute m hashed values per element scanned), and, worse, it would necessitate a large set of independent hashing functions, for which no construction is known [1].

The solution introduced in [15] under the name of *stochastic averaging*, consists in *emulating* the effect of m experiments with a *single hash function*. Roughly speaking, we divide the input stream $h(\mathcal{M})$ into m substreams, corresponding to a partition of the unit interval of hashed values into $[0, \frac{1}{m}[$,

Algorithm	*Cost*	(*units*)	*Accuracy*
Hit Counting [24]	$\frac{1}{10}N$	bits	$\approx$2%
Adaptive Sampling [12]	m	words ($\approx$32 bits)	$1.20/\sqrt{m}$
Probabilistic Counting [15]	m	words ($\leq$32 bits)	$0.78/\sqrt{m}$
MINCOUNT [2, 6, 16, 18]	m	words ($\leq$32 bits)	$1.00/\sqrt{m}$
LOGLOG [10]	m	bytes ($\leq$5 bits)	$1.30/\sqrt{m}$
HYPERLOGLOG	m	bytes ($\leq$5 bits)	$1.04/\sqrt{m}$

Fig. 1: A comparison of major cardinality estimators for cardinalities $\leq N$: (i) algorithms; (ii) memory complexity and units (for $N \leq 10^9$); (iii) relative accuracy.

$[\frac{1}{m}, \frac{2}{m}[, \ldots, [\frac{m-1}{m}, 1]$. Then, one maintains the m observables $O_1, \ldots, O_m$ corresponding to each of the m substreams. A suitable average of the $\{O_j\}$ is then expected to produce an estimate of cardinalities whose quality should improve, due to averaging effects, in proportion to $1/\sqrt{m}$, as m increases. The benefit of this approach is that it requires only a *constant number* of elementary operations per element of the multiset $\mathcal{M}$ (as opposed to a quantity proportional to m), while only *one hash function* is now needed.

The performances of several algorithms are compared in Figure 1; see also [13] for a review. HYPERLOGLOG, described in detail in the next section, is based on the same observable as LOGLOG, namely the largest ρ value obtained, where $\rho(x)$ is the position of the leftmost 1-bit in binary string x. Stochastic averaging in the sense above is employed. However, our algorithm differs from standard LOGLOG by its evaluation function: its is based on *harmonic means*, while the standard algorithm uses what amounts to a geometric mean[1]. The idea of using harmonic means originally drew its inspiration from an insightful note of Chassaing and Gérin [6]: such means have the effect of taming probability distributions with slow-decaying right tails, and here they operate as a variance reduction device, thereby appreciably increasing the quality of estimates. Theorem 1 below summarizes our main conclusions to the effect that the relative accuracy (technically, the *standard error*) of HYPERLOGLOG is numerically close to $\beta_\infty/\sqrt{m}$, where $\beta_\infty = \sqrt{3\log 2 - 1} \doteq 1.03896$. The algorithm needs to maintain a collection of registers, each of which is at most $\log_2 \log_2 N + O(1)$ bits, when cardinalities $\leq N$ need to be estimated. In particular, HYPERLOGLOG achieves an accuracy matching that of standard LOGLOG by consuming only 64% of the corresponding memory. As a consequence, using $m = 2048$, hashing on 32 bits, and short bytes of 5 bit length each: *cardinalities till values over $N = 10^9$ can be estimated with a typical accuracy of 2% using 1.5kB (kilobyte) of storage.*

The proofs base themselves in part on techniques that are now standard in analysis of algorithms, like poissonization, Mellin transforms, and saddle-point depoissonization. Some nonstandard problems however present themselves due to the special nonlinear character of harmonic means, so that several ingredients of the analysis are not completely routine.

1 The HYPERLOGLOG algorithm

The HYPERLOGLOG algorithm is fully specified in Figure 2, the corresponding program being discussed later, in Section 4. The input is a *multiset* $\mathcal{M}$ of data items, that is, a stream whose elements are read sequentially. The output is an estimate of the *cardinality*, defined as the number of distinct elements

[1] The paper [10] also introduces a variant called SUPERLOGLOG, which attempts to achieve variance reduction by censoring extreme data. It has however the disadvantage of not being readily amenable to analysis, as regards bias and standard error.

Let $h : \mathcal{D} \to [0,1] \equiv \{0,1\}^\infty$ *hash data from domain* $\mathcal{D}$ *to the binary domain.*
Let $\rho(s)$*, for* $s \in \{0,1\}^\infty$*, be the position of the leftmost 1-bit* ($\rho(0001\cdots) = 4$).
Algorithm HYPERLOGLOG (**input** $\mathcal{M}$: multiset of items from domain $\mathcal{D}$).
assume $m = 2^b$ with $b \in \mathbb{Z}_{>0}$;
initialize a collection of m registers, $M[1], \ldots, M[m]$, to $-\infty$;

for $v \in \mathcal{M}$ **do**
  set $x := h(v)$;
  set $j = 1 + \langle x_1 x_2 \cdots x_b \rangle_2$; {*the binary address determined by the first* b *bits of* x}
  set $w := x_{b+1} x_{b+2} \cdots$; **set** $M[j] := \max(M[j], \rho(w))$;
compute $Z := \left(\sum_{j=1}^{m} 2^{-M[j]} \right)^{-1}$; {*the "indicator" function*}
return $E := \alpha_m m^2 Z$ with α_m as given by Equation (3).

Fig. 2: The HYPERLOGLOG Algorithm.

in $\mathcal{M}$. A suitable hash function h has been fixed. The algorithm relies on a specific bit-pattern observable in conjunction with stochastic averaging. Given a string $s \in \{0,1\}^\infty$, let $\rho(s)$ represent the position of the leftmost 1 (equivalently one plus the length of the initial run of 0's). The stream $\mathcal{M}$ is split into substreams $\mathcal{M}_1, \ldots \mathcal{M}_m$, based on the first b bits of hashed values[2] of items, where $m = 2^b$, and each substream is processed independently. For $\mathcal{N} \equiv \mathcal{M}_j$ such a substream (regarded as composed of hashed values stripped of their initial b bits), the corresponding observable is then

$$\mathrm{Max}(\mathcal{N}) := \max_{x \in \mathcal{N}} \rho(x), \tag{1}$$

with the convention that $\mathrm{Max}(\emptyset) = -\infty$. The algorithm gathers on the fly (in registers $M[j]$) the values $M^{(j)}$ of $\mathrm{Max}(\mathcal{M}_j)$ for $j = 1 \ldots, m$. Once all the elements have been scanned, the algorithm computes the *indicator*,

$$Z := \left(\sum_{j=1}^{m} 2^{-M^{(j)}} \right)^{-1}. \tag{2}$$

It then returns a normalized version of the harmonic mean of the $2^{M^{(j)}}$ in the form,

$$E := \frac{\alpha_m m^2}{\sum_{j=1}^{m} 2^{-M^{(j)}}}, \qquad \text{with} \quad \alpha_m := \left(m \int_0^\infty \left(\log_2 \left(\frac{2+u}{1+u} \right) \right)^m du \right)^{-1}. \tag{3}$$

Here is the intuition underlying the algorithm. Let n be the unknown cardinality of $\mathcal{M}$. Each substream will comprise approximately n/m elements. Then, its Max-parameter should be close to $\log_2(n/m)$. The harmonic mean (mZ in our notations) of the quantities 2^{Max} is then likely to be of the order of n/m. Thus, $m^2 Z$ should be of the order of n. The constant α_m, provided by our subsequent analysis, is finally introduced so as to correct a systematic multiplicative bias present in $m^2 Z$.

Our main statement, Theorem 1 below, deals with the situation of *ideal multisets*:

[2] The algorithm can be adapted to cope with any integral value of $m \geq 3$, at the expense of a few additional arithmetic operations.

Definition 1 *An* ideal multiset *of cardinality n is a sequence obtained by arbitrary replications and permutations applied to n uniform identically distributed random variables over the real interval* $[0, 1]$.

In the analytical part of our paper (Sections 2 and 3), we postulate that the collection of hashed values $h(\mathcal{M})$, which the algorithm processes constitutes an ideal multiset. This assumption is a natural way to model the outcome of well designed hash functions. Note that the number of distinct elements of such an ideal multiset equals n with probability 1. We henceforth let $\mathbb{E}_n$ and $\mathbb{V}_n$ be the expectation and variance operators under this model.

Theorem 1 *Let the algorithm* HYPERLOGLOG *of Figure 2 be applied to an ideal multiset of (unknown) cardinality n, using $m \geq 3$ registers, and let E be the resulting cardinality estimate.*

(i) *The estimate E is* asymptotically almost unbiased *in the sense that*

$$\frac{1}{n}\mathbb{E}_n(E) \underset{n\to\infty}{=} 1 + \delta_1(n) + o(1), \text{ where } |\delta_1(n)| < 5 \cdot 10^{-5} \text{ as soon as } m \geq 16.$$

(ii) *The* standard error *defined as $\frac{1}{n}\sqrt{\mathbb{V}_n(E)}$ satisfies as $n \to \infty$,*

$$\frac{1}{n}\sqrt{\mathbb{V}_n(E)} \underset{n\to\infty}{=} \frac{\beta_m}{\sqrt{m}} + \delta_2(n) + o(1), \text{ where } |\delta_2(n)| < 5 \cdot 10^{-4} \text{ as soon as } m \geq 16,$$

the constants β_m being bounded, with $\beta_{16} \doteq 1.106$, $\beta_{32} \doteq 1.070$, $\beta_{64} \doteq 1.054$, $\beta_{128} \doteq 1.046$, and $\beta_\infty = \sqrt{3\log(2) - 1} \doteq 1.03896$.

The standard error measures in relative terms the typical error to be observed (in a mean quadratic sense). The functions $\delta_1(n), \delta_2(n)$ represent oscillating functions of a tiny amplitude, which are computable, and whose effect could in theory be at least partly compensated—they can anyhow be safely neglected for all practical purposes.

Plan of the paper. The bulk of the paper is devoted to the proof of Theorem 1. We determine the asymptotic behaviour of $\mathbb{E}_n(Z)$ and $\mathbb{V}_n(Z)$, where Z is the indicator $1/\sum 2^{-M^{(j)}}$. The value of α_m in Equation (3), which makes E an asymptotically almost unbiased estimator, is derived from this analysis, as is the value of the standard error. The mean value analysis forms the subject of Section 2. In fact, the exact expression of $\mathbb{E}_n(Z)$ being hard to manage, we first "poissonize" the problem and examine $\mathbb{E}_{\mathcal{P}(\lambda)}(Z)$, which represents the expected value of the indicator Z when the total number of elements is not fixed, but rather obeys a Poisson law of parameter λ. We then prove that, asymptotically, the behaviours of $\mathbb{E}_n(Z)$ and $\mathbb{E}_{\mathcal{P}(\lambda)}(Z)$ are close, when one chooses $\lambda := n$: this is the depoissonization step. The variance analysis of the indicator Z, hence of the standard error, is sketched in Section 3 and is entirely parallel to the mean value analysis. Finally, Section 4 examines how to implement the HYPERLOGLOG algorithm in real-life contexts, presents simulations, and discusses optimality issues.

2 Mean value analysis

Our starting point is the random variable Z (the "indicator") defined in (2). We recall that $\mathbb{E}_n$ refers to expectations under the ideal multiset model, when the (unknown) cardinality n is fixed. The analysis starts from the exact expression of $\mathbb{E}_n(Z)$ in Proposition 1, continues with an asymptotic analysis of the corresponding Poisson expectation summarized by Proposition 2, and concludes with the depoissonization argument of Proposition 3.

2.1 Exact expressions

Let $\mathcal{N}$ be an ideal multiset of cardinality ν. The quantity $\mathrm{Max}(\mathcal{N}) = \max_{x\in\mathcal{N}} \rho(x)$ is the maximum of ν independent random variables, namely, the values $\rho(x)$. Each random variable, call it Y, is geometrically distributed according to $\mathbb{P}(Y \geq k) = 2^{1-k}$, for $k \geq 1$. Thus, the maximum M of ν such random variables satisfies $\mathbb{P}(M = k) = \left(1 - \frac{1}{2^k}\right)^\nu - \left(1 - \frac{1}{2^{k-1}}\right)^\nu$, for $\nu \geq 1$. Let now an ideal multiset of *fixed* cardinality n be split into m "submultisets" of respective (random) cardinalities $N^{(1)}, \ldots, N^{(m)}$. The joint law of the $N^{(j)}$ is a multinomial. The combination of the previous two observations then provides:

Proposition 1 *The expectation of the indicator Z resulting from an ideal multiset of* fixed *cardinality n satisfies*

$$\mathbb{E}_n(Z) = \sum_{k_1,\ldots,k_m\geq 1} \frac{1}{\sum_{j=1}^m 2^{-k_j}} \sum_{n_1+\ldots+n_m=n} \binom{n}{n_1,\ldots,n_m} \frac{1}{m^n} \prod_{j=1}^m \gamma_{n_j,k_j}, \quad \textit{where } \gamma_{\nu,k} = (1-\frac{1}{2^k})^\nu - (1-\frac{1}{2^{k-1}})^\nu, \tag{4}$$

for $\nu, k \geq 1$ and $\gamma_{0,k} = 0$.

Note that, under the convention that registers $M^{(j)}$ are initialized to $-\infty$, we have $Z = 0$ as soon as *any* of the registers has remained untouched—this explains the fact that summation in (4) only needs to be taken over register values $k_j \geq 1$.

The rather formidable looking expression of (4) is to be analysed. For this purpose, we introduce the Poisson model, where an ideal multiset is produced with a *random* size N distributed according to a Poisson law of parameter λ: $\mathbb{P}(N = n) = e^{-\lambda}\lambda^n/n!$. Then, as shown by a simple calculation, we have: *Under the Poisson model of rate λ, the expectation of the indicator Z satisfies,*

$$\mathbb{E}_{\mathcal{P}(\lambda)}(Z) = \sum_{k_1,\ldots,k_m\geq 1} \frac{1}{\sum_{j=1}^m 2^{-k_j}} \prod_{j=1}^m g\left(\frac{\lambda}{m2^{k_j}}\right), \quad \textit{where} \quad g(x) = e^{-x} - e^{-2x}. \tag{5}$$

The verification is based on the relation,

$$\mathbb{E}_{\mathcal{P}(\lambda)}(Z) = \sum_{n\geq 0} \mathbb{E}_n(Z) e^{-\lambda} \frac{\lambda^n}{n!},$$

and series rearrangements. (Equivalently, independence properties of Poisson flows may be used.)

2.2 Asymptotic analysis under the Poisson model

The purpose of this subsection is to determine the asymptotic behaviour of the Poisson expectation, $\mathbb{E}_{\mathcal{P}(\lambda)}(Z)$, as given by Equation (5). Our main result in this subsection is summarized in the following proposition:

Proposition 2 *With α_m as in* (3)*, the Poisson expectation $\mathbb{E}_{\mathcal{P}(\lambda)}(Z)$ satisfies*

$$\mathbb{E}_{\mathcal{P}(\lambda)}(Z) \underset{\lambda\to\infty}{=} \frac{\lambda}{m}\left(\frac{1}{m\alpha_m} + \epsilon_m\left(\frac{\lambda}{m}\right) + o(1)\right), \quad \textit{where } |\epsilon_m(t)| < 5\cdot 10^{-5}/m \textit{ as soon as } m \geq 16. \tag{6}$$

The proof of Proposition 2 consists of three steps: (i) the Poisson expectation is first expressed in integral form (Equation (9)); (ii) the integrand is next analysed by means of the Mellin transform (Lemma 1); (iii) the outcome of the local Mellin analysis is finally used to estimate the Poisson expectation.

The integral representation. The asymptotic analysis of the Poisson expectation departs from the usual paradigm of analysis exemplified by [10, 14, 23] because of the coupling introduced by the harmonic mean, namely, the factor $(\sum 2^{-k_j})^{-1}$. This is remedied by a use of the simple identity

$$\frac{1}{a} = \int_0^\infty e^{-at}\, dt, \tag{7}$$

which then leads to a crucial separation of variables,

$$\begin{aligned} \mathbb{E}_{\mathcal{P}(mx)}(Z) &= \sum_{k_1,\ldots,k_m \geq 1} \frac{1}{\sum_{j=1}^m 2^{-k_j}} \prod_{j=1}^m g\left(\frac{x}{2^{k_j}}\right) \\ &= \sum_{k_1,\ldots,k_m \geq 1} \int_0^\infty \prod_{j=1}^m g(2^{-k_j}x) e^{-t\sum_{j=1}^m 2^{-k_j}}\, dt \quad = \quad \int_0^\infty G(x,t)^m\, dt, \end{aligned}$$

where we have set

$$G(x,t) := \sum_{k\geq 1} g\left(\frac{x}{2^k}\right) e^{-t/2^k}. \tag{8}$$

Then the further change of variables $t = xu$ leads to the following useful form: *The Poisson expectation satisfies, with $G(x,t)$ defined by* (8):

$$\mathbb{E}_{\mathcal{P}(x)}(Z) = H\left(\frac{x}{m}\right), \qquad \text{where} \quad H(x) := x\int_0^{+\infty} G(x,xu)^m\, du. \tag{9}$$

Analysis of the integrand. Our goal is now to analyse the integral representation (9) of Poisson averages. We make use of the Mellin transform, which to a function $f(t)$ defined on $\mathbb{R}_{>0}$, associates the complex function

$$f^\star(s) := \int_0^{+\infty} f(t)t^{s-1}\, dt. \tag{10}$$

One fundamental property is that *the transform of a harmonic sum, $F(x) = \sum_k \lambda_k f(\mu_k x)$, factorizes*, as $F^\star(s) = \left(\sum_k \lambda_k \mu_k^{-s}\right) f^\star(s)$. Another fundamental property (devolving from the inversion formula and a residue calculation) is that *the asymptotic behaviour of the original function, f, can be read off from the singularities of its transform $f^\star$;* see [14] for a survey. We prove:

Lemma 1 *For each fixed $u > 0$, the function $x \mapsto G(x,xu)$ has the following asymptotic behaviour as $x \to +\infty$,*

$$G(x,xu) = \begin{cases} f(u)\left(1 + O(x^{-1})\right) + u\epsilon(x,u) \text{ if } u \leq 1 \\ f(u)(1 + 2\widetilde{\epsilon}(x,u) + O(x^{-1})) \text{ if } u > 1. \end{cases}, \tag{11}$$

where $f(u) = \log_2\left(\frac{2+u}{1+u}\right)$, the $\mathcal{O}$ error terms are uniform in $u > 0$, and $|\epsilon|, |\widetilde{\epsilon}| \leq \epsilon_0 \simeq 7\cdot 10^{-6}$ for $x \geq 0$.

Proof: Write $h_u(x) := G(x,xu)$. This function is a harmonic sum,

$$h_u(x) = \sum_{k=1}^{+\infty} g(2^{-k}x)e^{-2^{-k}xu} = \sum_{k=1}^{+\infty} q(x2^{-k}), \text{ with } q(x) := g(x)e^{-xu},$$

so that its Mellin transform factorizes, in the fundamental strip $\langle -1, 0\rangle$, as

$$h_u^\star(s) = \left(\sum_{k=1}^{+\infty} 2^{ks}\right) q^\star(s) = \frac{2^s\Gamma(s)}{1-2^s}\left((1+u)^{-s} - (2+u)^{-s}\right), \tag{12}$$

where $\Gamma(s)$ is the Euler Gamma function. The asymptotic behaviour of $h_u(x)$ as $x \to +\infty$ is then determined by the poles of $h_u^\star(s)$ that lie on the right of the fundamental strip. The poles of $h_u^\star(s)$ are at $\mathbb{Z}_{<0}$ (because of the Γ factor) and at the complex values $\{\eta_k := 2ik\pi/\log(2),\ k \in \mathbb{Z}\}$, where the denominator $1-2^s$ vanishes. The Mellin inversion formula,

$$h_u(x) = \frac{1}{2i\pi}\int_{-1/2-i\infty}^{-1/2+i\infty} h_u^\star(s) x^{-s}\, ds,$$

when combined with the residue theorem, implies

$$h_u(x) = -\sum_{k\in\mathbb{Z}} \mathrm{Res}(h_u^\star(s)x^{-s}, \eta_k) + \frac{1}{2i\pi}\int_{1-i\infty}^{1+i\infty} h_u^\star(s)x^{-s}\, ds. \tag{13}$$

Some care is to be exerted in estimations since *uniformity* with respect to the parameter u is required.

The residues are given by

$$\begin{cases} \mathrm{Res}\left(h_u^\star(s)x^{-s}, \eta_k\right) = \dfrac{1}{\log 2} x^{-\eta_k}\Gamma(\eta_k)\left((1+u)^{-\eta_k} - (2+u)^{-\eta_k}\right) & (k \in \mathbb{Z}_{\neq 0}) \\ \mathrm{Res}\left(h_u^\star(s)x^{-s}, 0\right) = \dfrac{1}{\log 2}\log\left(\dfrac{2+u}{1+u}\right) & (k=0). \end{cases}$$

As regards their sum in (13), we note the inequality, valid for $\Re(s) \geq 0$,

$$|(1+u)^{-s} - (2+u)^{-s}| = |e^{-s\log(1+u)} - e^{-s\log(2+u)}| \leq |s|\left|\log\left(\frac{2+u}{1+u}\right)\right| \tag{14}$$

(verified by writing the difference as the integral of its derivative, then bounding the derivative), and its companion, valid for $s = \eta_k$ (to be used for u close to 0)

$$\left|\left((1+u)^{-s} - (2+u)^{-s}\right)\right| \leq \frac{1}{2}|s|u, \tag{15}$$

verified by the strict decrease of $u \mapsto |(1+u)^{-s} - (2+u)^{-s}|/u$. As a consequence, one obtains the two simultaneously valid bounds

$$\left|\sum_{k\in\mathbb{Z}} \mathrm{Res}(h_u^\star(s)x^{-s}, \eta_k) - \log_2\left(\frac{2+u}{1+u}\right)\right| \leq \begin{cases} \dfrac{\pi}{(\log 2)^2} u \displaystyle\sum_{k\in\mathbb{Z}, k\neq 0} |k\Gamma(\eta_k)| \\ \dfrac{2\pi}{(\log 2)^2}\log\left(\dfrac{2+u}{1+u}\right)\displaystyle\sum_{k\in\mathbb{Z}, k\neq 0} |k\Gamma(\eta_k)|, \end{cases}$$

out of which one extracts, with $\epsilon_0 \simeq 7 \cdot 10^{-6}$:

$$\left|\sum_{k\in\mathbb{Z}} \mathrm{Res}(h_u^\star(s)x^{-s}, \eta_k) - \log_2\left(\frac{2+u}{1+u}\right)\right| \le \begin{cases} \epsilon_0 u & \text{if } u \le 1 \\ 2\epsilon_0 \log\left(\frac{2+u}{1+u}\right) & \text{if } u > 1. \end{cases} \tag{16}$$

Next, we turn to the integral remainder in (13). By the inequality of (14), one has uniformly

$$\left|\int_{1-i\infty}^{1+i\infty} h_u^\star(s)x^{-s}ds\right| \le \frac{1}{x}\log\left(\frac{2+u}{1+u}\right)\int_{1-i\infty}^{1+i\infty} |s\Gamma(s)|\,d|s| = O\left(\frac{1}{x}\right)\log\left(\frac{2+u}{1+u}\right). \tag{17}$$

The two bounds (16) and (17) then justify the statement. ∎

Final asymptotics of the Poisson averages. There now remains to estimate the function $H(x)$ of (9), with Lemma 1 providing precise information on the integrand. Accordingly, we decompose the domain of the integral expressing $H(x)$,

$$\frac{1}{x}H(x) = \int_0^1 (f(u) + u\epsilon(x,u))^m\,du + \int_1^\infty f(u)^m(1 + 2\widetilde{\epsilon}(x,u))^m\,du + o(1) \equiv A + B + o(1),$$

with $A = \int_0^1$ and $B = \int_1^\infty$. Here, like before, we have set $f(u) := \log_2\left(\frac{2+u}{1+u}\right)$.

We first estimate A by means of the inequality $f(u) \le 1 - 2u/5$, for $u \in [0,1]$:

$$\begin{aligned}
\left|A - \int_0^1 f(u)^m\,du\right| &\le \sum_{k=1}^m \binom{m}{k}\int_0^1 (1-2u/5)^{m-k}(u\epsilon_0)^k du \\
&= \int_0^1 (1-2u/5+u\epsilon_0)^m - (1-2u/5)^m du \\
&= \frac{1}{m+1}\left(a(\epsilon_0) - a(0)\right), \quad \text{where } a(v) = \frac{1-(3/5+v)^{m+1}}{2/5-v}.
\end{aligned}$$

Thus, upon bounding $a'(v)$ near 0, one finds:

$$\left|A - \int_0^1 (f(u))^m\,du\right| \le \frac{6.26\,\epsilon_0}{m+1}. \tag{18}$$

As regards the estimation of B, it suffices to note that, for $u \ge 1$, one has $f(u) \le 1/((1+u)\log 2)$ in order to get directly:

$$\left|B - \int_1^\infty f(u)^m\,du\right| \le \frac{2}{m-1}\left(\frac{(1+2\epsilon_0)^m - 1}{(2\log(2))^m}\right). \tag{19}$$

The combination of (18) and (19) finally give us

$$H(x) = x\left(\int_0^\infty f(u)^m\,du + \epsilon_m(x) + o(1)\right), \tag{20}$$

where $|\epsilon_m(x)| < 5 \cdot 10^{-5}/m$ (for $m \ge 16$). This last estimate applied to the expression (9) of Poisson averages then concludes the proof of Proposition 2.

2.3 Analysis under the fixed-size model (depoissonization)

We can now conclude the average-case analysis of the main indicator Z by showing that the asymptotic approximation derived for the Poisson model (Proposition 2) applies to the fixed size model, up to negligible error terms. To this aim, we appeal to a technique known as "*analytic depoissonization*", pioneered by Jacquet and Szpankowski (see [19] and [23, p. 456]) and based on the saddle point method. To wit:

Theorem (Analytic depoissonization). *Let $f(z) = e^{-z} \sum f_k z^k/k!$ be the Poisson generating function of a sequence (f_k). Assume $f(z)$ to be entire. Assume also that there exists a cone $S_\theta = \{z \,/\, z = re^{i\phi},\ |\phi| \le \theta\}$ for some $\theta < \frac{\pi}{2}$ and a real number $\alpha < 1$ such that the following two conditions are satisfied, as $|z| \to \infty$:*

$\mathbf{C_1}$*: for $z \in S_\theta$, one has $|f(z)| = O(|z|)$,*

$\mathbf{C_2}$*: for $z \notin S_\theta$, one has $|f(z)e^z| = O(e^{\alpha|z|})$.*

Then $f_n = f(n) + O(1)$.

The use of this theorem amounts to estimating Poisson averages (the quantity $f(z)$), *when the Poisson rate $\lambda \equiv z$ is allowed to vary in the complex plane*, in which case it provides a way to return asymptotically to fixed-size estimates (the sequence f_n). The Mellin technology turns out to be robust enough to allow for such a method to be used.

Proposition 3 *The expectation of the mean value of the* HYPERLOGLOG *indicator Z applied to a multiset of* fixed *cardinality n satisfies asymptotically as $n \to \infty$*

$$\mathbb{E}_n(Z) = \mathbb{E}_{\mathcal{P}(n)}(Z) + O(1).$$

Proof: We apply analytic depoissonization to the integral expression (9), which we repeat here:

$$\mathbb{E}_{\mathcal{P}(x)}(Z) = H\left(\frac{x}{m}\right), \qquad \text{where} \quad H(x) := \int_0^\infty G(x,t)^m\, dt = x \int_0^{+\infty} G(x,xu)^m\, du. \tag{21}$$

We shall check the conditions $\mathbf{C_1}, \mathbf{C_2}$ of analytic depoissonization, for $f(z) := H(z/m)$, choosing the half-angle of the cone to be $\theta = \pi/3$ and $\alpha = 3/5$.

Inside the cone (Condition $\mathbf{C_1}$). It is sufficient to establish that $H(z) = O(|z|)$. The bounds are obtained via the second integral form of (21) by suitably revisiting the Mellin analysis of Subsection 2.2. Start from Equation (13), which expresses the Poisson expectation as a sum of residues, plus a remainder integral. When x is assigned a complex value $z = re^{i\phi}$, the quantity z^{-s} satisfies, for $s = \sigma + i\tau$:

$$z^{-s} = \left(r^{-\sigma} e^{\tau\phi}\right) \cdot \left(r^{-i\tau} e^{-i\phi\sigma}\right).$$

There, the second factor has modulus equal to 1, while the first remains $O(r^{-\sigma} e^{\pi|\tau|/3})$ within the cone. Then, given the fast decay of $\Gamma(s)$ towards $\pm i\infty$, the series of residues in (13) is still bounded and the remainder integral is itself $O(1)$. Thus the main estimate stated in Lemma 1 and relative to $h_u(x) \equiv G(x,xu)$ remains valid for $x = z$ within the cone, and the argument can be extended to show that the asymptotic form of $H(x)$ also holds (only the numerical bounds on the amplitude of the fluctuations need to be weakened). As a consequence, one has $H(z) = O(|z|)$, hence $H(z/m) = O(|z|)$ within the cone.

Outside the cone (Condition $\mathbf{C_2}$*).* Start from the first integral in (21). We subdivide the domain according to $\Re(z) < 0$ and $\Re(z) \geq 0$. For the case $\Re(z) < 0$, we set $L := \lfloor \log_2(|z|/m) \rfloor$. The definition of G in (8) gives

$$G\left(\frac{z}{m}, t\right) \quad = \quad \sum_{k \leq L} g\left(\frac{z}{m2^k}\right) e^{-t/2^k} + \sum_{k > L} g\left(\frac{z}{m2^k}\right) e^{-t/2^k},$$

so that

$$\left|G\left(\frac{z}{m}, t\right)\right| \quad \leq \quad \sum_{k \leq L} 2\left|e^{-z/m}\right| e^{-t/2^L} + C \sum_{k > L} \left|\frac{z}{m2^k}\right| e^{-t/2^k},$$

for some $C > 0$. Hence, raising to the mth power, we get

$$\left|G\left(\frac{z}{m}, t\right)\right|^m \leq 4^m L \left|e^{-z} e^{-m^2 t/|z|}\right| + 2^m C^m \frac{|z|^m}{m^m} \left(\sum_{k \geq 1} \frac{1}{2^k} e^{-t/2^k}\right),$$

where use has been made of the inequality $(X+Y)^m \leq 2^m X^m + 2^m Y^m$, valid for arbitrary $X, Y \geq 0$. Now, since the integral

$$\Gamma_m := \int_0^\infty \left(\sum_{k \geq 1} \frac{1}{2^k} e^{-t/2^k}\right)^m dt \tag{22}$$

converges, we obtain, upon integrating,

$$\left|H\left(\frac{z}{m}\right)\right| \leq \int_0^{+\infty} \left|G\left(\frac{z}{m}, t\right)\right|^m dt \leq A\left|e^{-z} z \log_2 |z|^m\right| + B\left|z^m\right|,$$

where A and B are constants (depending on m). Consequently, for $\Re(z) < 0$, we have $|e^z H(z/m)| = O(e^{\alpha|z|})$ for any $\alpha > 0$, and in particular we may adopt $\alpha = \frac{3}{5}$.

For the case $\Re(z) \geq 0$, it suffices to note that

$$\left|G\left(\frac{z}{m}, t\right)\right| \leq |z| \sum_k \frac{1}{2^k} e^{-t/2^k}.$$

Raising to the mth power and integrating, we get

$$\left|H\left(\frac{z}{m}\right)\right| \leq \int_0^\infty \left|G\left(\frac{z}{m}, t\right)\right|^m dt \leq |z|^m \Gamma_m,$$

with Γ_m as in (22), so that

$$\left|e^z H\left(\frac{z}{m}\right)\right| = O(|e^z z^m|) = O(e^{|z|/2} z^m) = O(e^{3|z|/5}),$$

since we have $\Re(z) < \frac{1}{2}|z|$ outside the cone. This last inequality completes the proof of the statement. ∎

The proof of the unbiased character of HYPERLOGLOG, corresponding to Part (i) of our main Theorem 1, is thus essentially complete: it suffices to combine Propositions 2 and 3 giving the asymptotic estimation of the expectation $\mathbb{E}_n(Z)$ of the indicator, in order to get the expected value of the estimator $\mathbb{E}_n(E) \approx n$ via a simple normalization by the constant factor $m^2 \alpha_m$.

3 Variance and other stories

3.1 Variance analysis

The estimation of the variance of the indicator, namely $\mathbb{V}_n(Z) = \mathbb{E}_n(Z^2) - \mathbb{E}_n^2(Z)$, serves to justify Part (ii) of our main Theorem 1, and hence characterizes the accuracy of HYPERLOGLOG. Since the analysis develops along lines that are entirely parallel to those of Section 2, we content ourselves with a brief indication of the main steps of the proof.

The starting point is an expression of the moment of order 2 of the indicator Z under the Poisson model,

$$\mathbb{E}_{\mathcal{P}(\lambda)}(Z^2) = \sum_{k_1,\ldots,k_m \geq 1} \left(\frac{1}{\sum_{j=1}^{m} 2^{-k_j}} \right)^2 \prod_{j=1}^{m} g\left(\frac{\lambda}{m 2^{k_j}}\right), \tag{23}$$

which is the analogue of (5). Then, the use of the identity

$$\frac{1}{a^2} = \int_0^\infty t e^{-at} dt$$

leads to the integral form (compare with (9))

$$\mathbb{E}_{\mathcal{P}(\lambda)}(Z^2) = K\left(\frac{\lambda}{m}\right), \qquad \text{where} \quad K(x) := x^2 \int_0^{+\infty} u G(x, xu)^m \, du. \tag{24}$$

The integral being very close to the one that represents $\mathbb{E}_{\mathcal{P}(\lambda)}(Z)$, the analysis of the integrand available from Lemma 1 can be entirely recycled. We then find

$$K(x) = x^2 \left(\int_0^\infty u f(u)^m \, du + \epsilon'_m(x) + o(1) \right), \tag{25}$$

where $\epsilon'(x)$ is a small oscillating function, implying

$$\mathbb{V}_{\mathcal{P}(\lambda)}(Z) = \lambda^2 \left(\int_0^\infty u f(u)^m \, du - \left(\int_0^\infty f(u)^m \, du \right)^2 + \epsilon''(\lambda) + o(1) \right) \tag{26}$$

(with ϵ'' small), which constitutes the analogue of Proposition 2.

The last estimate (26) can then be subjected to depoissonization (with a proof similar to that of Proposition 3), to the effect that

$$\mathbb{V}_n(Z) = \mathbb{V}_{\mathcal{P}(n)}(Z) + O(n). \tag{27}$$

This shows that the standard error, measured by $\frac{1}{n}\sqrt{\mathbb{V}_n(E)}$, is, for each fixed m, asymptotic to a constant as $n \to \infty$, neglecting as we may tiny fluctuations. The stronger property that this constant is of the form $\beta_m/\sqrt{m}$ (with β_m bounded) is established in the next subsection.

3.2 Constants

There only remains to discuss the proportionality constants that determine the shapes of the bias-correction constant α_m specified in (3) and of the standard-error constant β_m of the statement of Theorem 1. Define the special integrals

$$J_s(m) = \int_0^\infty u^s f(u)^m \, du.$$

We have from Equations (3) and (26)

$$\alpha_m = \frac{1}{mJ_0(m)}, \qquad \beta_m = \sqrt{m}\sqrt{\frac{J_1(m)}{J_0(m)^2} - 1}.$$

The integrals $J_s(m)$ are routinely amenable to the Laplace method [8]:

$$\begin{cases} J_0(m) &= \dfrac{2\log 2}{m}\left(1 + \dfrac{1}{m}(3\log 2 - 1) + O(m^{-2})\right) \\ J_1(m) &= \dfrac{(2\log 2)^2}{m^2}\left(1 + \dfrac{3}{m}(3\log 2 - 1) + O(m^{-2})\right). \end{cases} \tag{28}$$

Thus the bias correction α_m and the variance constant β_m satisfy

$$\alpha_m \sim \frac{1}{2\log 2} \doteq 0.72134, \qquad \beta_m \sim \sqrt{3\log 2 - 1} \doteq 1.03896 \qquad (m \to +\infty), \tag{29}$$

which turn out to provide good numerical approximations, even for relatively low values of m. These estimates imply in particular that β_m remains bounded for all $m \geq 3$, which concludes the proof of Theorem 1.

Additionally, we observe that the constants α_m, β_m belong to an interesting arithmetic class: *the integrals $J_0(m), J_1(m)$ are expressible as rational combinations of $L = \log 2$, values of the Riemann zeta function at the integers, and polylogarithms evaluated at $\frac{1}{2}$.* For instance: $J_0(2) = \frac{\pi^2}{6L^2} - 2$, $J_0(3) = \frac{3\zeta(3)}{4L^3} - 2$, and

$$J_0(4) = \frac{1}{L^4}\left(\frac{4}{15}\pi^4 - \pi^2 L^2 - 3L^4 - 21L\zeta(3) - 24\,\mathrm{Li}_4(\frac{1}{2})\right), \qquad \text{where} \qquad \mathrm{Li}_r(z) := \sum_{n\geq 1}\frac{z^n}{n^r}.$$

They can thereby be computed to great accuracy.

4 Discussion

We offer here final reflections concerning an implementation of the HYPERLOGLOG algorithm (Figures 3, 4, and 5) as well as some surrounding complexity considerations.

The* HYPERLOGLOG *program. A *program* meant to cope with most practical usage conditions is described in Figure 3. In comparison to the *algorithm* of Figure 2, one modification regarding initialization and two final corrections to the estimates are introduced.

(i) *Initialization of registers.* In the algorithm of Figure 2, registers are initialized at $-\infty$. This has the advantage of leading to expressions of the average-case that are comparatively simple: see Equations (4) and (5). However, a consequence is that the estimate E returned by the algorithm assumes the value 0 as soon as one of the registers has been left untouched, that is, as soon as one of the m substreams is empty. Given known fact regarding the coupon collector problem, this means that we should expect $E = 0$ when $n \ll m\log m$, so that the algorithm errs badly for small cardinalities. In the program of Figure 3, we have changed the initialization of registers to 0. The conclusions

Let $h : \mathcal{D} \to \{0,1\}^{32}$ *hash data from* $\mathcal{D}$ *to binary 32–bit words.*
Let $\rho(s)$ *be the position of the leftmost 1-bit of s: e.g.,* $\rho(1\cdots) = 1,\ \ \rho(0001\cdots) = 4,\ \ \rho(0^K) = K+1$.
define $\alpha_{16} = 0.673;\quad \alpha_{32} = 0.697;\quad \alpha_{64} = 0.709;\quad \alpha_m = 0.7213/(1+1.079/m)$ for $m \geq 128$;

Program HYPERLOGLOG (**input** $\mathcal{M}$: multiset of items from domain $\mathcal{D}$).
assume $m = 2^b$ with $b \in [4\,..\,16]$.
initialize a collection of m registers, $M[1], \ldots, M[m]$, to 0;

for $v \in \mathcal{M}$ **do**
  set $x := h(v)$;
  set $j = 1 + \langle x_1 x_2 \cdots x_b \rangle_2$; {*the binary address determined by the first* b *bits of* x}
  set $w := x_{b+1} x_{b+2} \cdots$;
  set $M[j] := \max(M[j], \rho(w))$;
compute $E := \alpha_m m^2 \cdot \left(\sum_{j=1}^{m} 2^{-M[j]} \right)^{-1}$; {*the "raw" HyperLogLog estimate*}
if $E \leq \frac{5}{2}m$ **then**
  let V be the number of registers equal to 0;
  if $V \neq 0$ **then set** $E^\star := m \log(m/V)$ **else set** $E^\star := E$; {*small range correction*}
if $E \leq \frac{1}{30}2^{32}$ **then**
  set $E^\star := E$; {*intermediate range—no correction*}
if $E > \frac{1}{30}2^{32}$ **then**
  set $E^\star := -2^{32} \log(1 - E/2^{32})$; {*large range correction*}
return *cardinality estimate* $E^\star$ *with typical relative error* $\pm 1.04/\sqrt{m}$.

Fig. 3: The HYPERLOGLOG Program dimensioned for maximal cardinalities in the range $[0\,..\,10^9]$ and for common "practical" values $m = 2^4, \ldots, 2^{16}$.

of Theorem 1, regarding the *asymptotically* unbiased character of the estimate, are still applicable to the program, since all substreams are nonempty with an overwhelming probability, as soon as $n \gg m \log m$. The advantage of the modification is that we can now get usable estimates even when n is a small multiple of m (this fact can be furthermore confirmed by Poisson approximations). The estimates provided by the program for very small values of n (say, n a constant or $n = m$) can then be effectively corrected, as we explain next.

(ii) *Small range corrections.* For the HYPERLOGLOG program (including the modification of (i) above, regarding register initialization), extensive simulations demonstrate that the asymptotic regime is practically attained (without essentially affecting the nominal error of $1.04/\sqrt{m}$ and without detectable bias) at the cardinality value $n = \frac{5}{2}m$, when $m \geq 16$. In contrast, for $n < \frac{5}{2}m$, nonlinear distortions start appearing—on the extreme side, the raw algorithm with registers initialized to 0 will invariably return the estimate $\alpha_m m \doteq 0.7m$ when $n = 0$ (!). Thus, corrections *must* be brought to the estimate, when E (i.e., n) is comparatively small with respect to m.

The solution comes from probabilistic properties of random allocations, as already exploited by the HIT COUNTING algorithm of Whang *et al.* [24], whose analysis is discussed in [9, Sec. 4.3]. Say n balls are thrown at random into m bins. Then, as it is well-known, the number of empty bins is about $me^{-\mu}$, where $\mu := n/m$. Thus, upon observing V empty bins amongst a total of m, one may legitimately expect μ to be close to $\log(m/V)$, that is, n must be close to $m\log(m/V)$. (The quality of this estimate can be precisely analysed, since exact and asymptotic forms are known for the mean, variance, and distribution of V; see, e.g., [21].) Here the bins are the ones associated to the m "submultisets", and one knows that a bin j is empty from the fact that its corresponding register $M[j]$ has preserved its initial value 0. This correction is incorporated in the program of Figure 3.

(iii) *Large range corrections.* For cardinalities in the range $1\,..\,N$, with N of the order of 10^9, hashing over at least $L = 32$ bits should be used ($2^{32} \doteq 4 \cdot 10^9$). However, when the cardinality n approaches (or perhaps even exceeds) 2^L, then hashing collisions become more and more likely. For a randomly chosen hash function, this effect can be modelled by a balls and bins model of the type described in the previous paragraph, with now 2^L replacing m. In other words, the quantity E of HYPERLOGLOG estimates the number of different hashed values, which is with high probability, about $2^L(1-e^{-\Lambda})$, where $\Lambda = n/2^L$. The inversion of that relation then gives us the approximate equation $n = -2^L\log(1-E/2^L)$, which is the one used in the program.

Regarding registers, their values *a priori* range in the interval $0\,..\,L+1-\log_2 m$. With hashed values of 32 bits, this means that 5 bits ("short bytes") are sufficient to store registers (of course, standard 8–bit bytes can also be used in some implementations). Regarding the quality of results returned, we expect the values of the estimate returned to be approximately Gaussian, due to an averaging effect and the Central Limit Theorem: this property is indeed well supported by the simulations of Figure 4 (bottom). Accordingly:

Let $\sigma \approx 1.04/\sqrt{m}$ represent the standard error; the estimates provided by HYPERLOGLOG *are expected to be within σ, 2σ, 3σ of the exact count in respectively 65%, 95%, 99% of all the cases.*

In practice, the HYPERLOGLOG program is quite efficient: like the standard LOGLOG of [10] or MINCOUNT of [16], its running time is dominated by the computation of the hash function, so that it is only three to four times slower than a plain scan of the data (e.g., by the Unix command "`wc -l`", which merely counts end-of-lines).

Optimality considerations. The near-optimality expressed by our title results from the combination of two facts.

(i) Clearly, maintaining ϵ-approximate counts till a range of N necessitates $\Omega(\log\log N)$ bits. Indeed, the cardinalities should be located in an exponential scale,

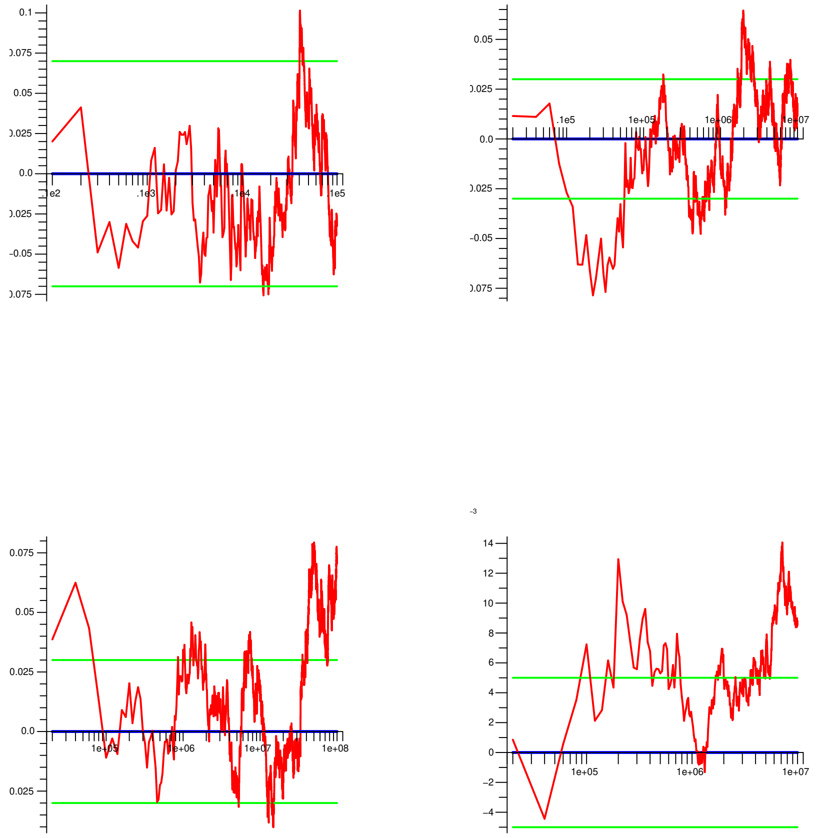
$$1, \quad (1+\epsilon), \quad (1+\epsilon)^2, \quad \cdots \quad , \quad (1+\epsilon)^L = N,$$

which comprises $\log_{(1+\epsilon)} N$ intervals, necessitating at least $\log_2\log_{(1+\epsilon)} N$ bits of information to be represented.

(ii) For a wide class of algorithms based on order statistics, Chassaing and Gérin [6] have shown that the best achievable accuracy is bounded from below by a quantity close to $1/\sqrt{m}$. Our algorithm, which can be viewed as maintaining approximate order statistics[3] is, on the basis of this result only

[3] In effect, for a multiset S of [0, 1]–numbers, the quantity $2^{-\max_S(\rho(x))}$ is an approximation to $\min(S)$ up to a factor at most 2.

• Traces of four typical executions showing the evolution over ideal multisets of the relative error of the estimate produced by the HYPERLOGLOG program as a function of cardinality n, for $n \le N$ and for various values of $\langle N, m\rangle$: (top left) $\langle 10^4, 256\rangle$; (top right) $\langle 10^7, 1024\rangle$; (bottom left) $\langle 10^8, 1024\rangle$; (bottom right) $\langle 10^7, 65536\rangle$. Values of n are plotted on a logarithmic scale.



(The top and botton horizontal lines represent the predicted standard error, namely, $\pm 7\%$ for $m = 256$, $\pm 3\%$ for $m = 1024$, and $\pm 0.5\%$ for $m = 65536$.)

• The empirical histogram of the estimates produced by the algorithm (based on 500 and 250 simulations, respectively) and an approximate fit by a Gaussian curve for $(n, m) = (10^4, 256)$, left, or $(n, m) = (10^6, 1024)$, right.

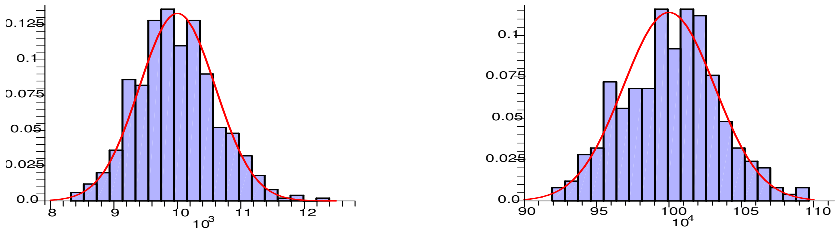


Fig. 4: Simulations of the behaviour of the HYPERLOGLOG program, including the low cardinality correction, on ideal multisets (random uniform data).

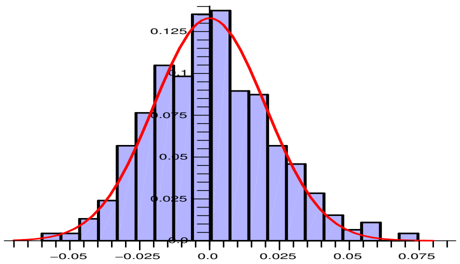


Fig. 5: Empirical histogram of the quality of the estimates measured by the relative errors observed, as provided by HyperLogLog on "real-life" files. The parameter is $m = 2048$, corresponding to a standard error of $\pm 2\%$; the experiment has been conducted on 458 chunks of about 40,000 lines each, obtained from octal dumps of the postscript source of a forthcoming book (*Analytic Combinatorics* by Flajolet and Sedgewick).

about 4% off the information-theoretic optimum of the Chassaing-Gérin class, while using memory units that are typically 3 to 5 times shorter.

As a final summary, the algorithm proves to be easy to code and efficient, being even nearly optimal under certain criteria. On "real-life" data, it appears to be in excellent agreement with the theoretical analysis, a fact recently verified by extensive tests (see Figure 5 for a sample) conducted by Pranav Kashyap, whose contribution is here gratefully acknowledged. The program can be applied to very diverse collections of data (only a "good" hash function is needed), and, once duly equipped with corrections, it can smoothly cope with a wide range of cardinalities–from *very* small to *very* large. In addition, it parallelizes or distributes[4] optimally and can be adapted to the "sliding window" usage [7].

All in all, HYPERLOGLOG is highly practical, versatile, and it conforms well to what analysis predicts.

References

[1] ALON, N., MATIAS, Y., AND SZEGEDY, M. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences 58*, 1 (1999), 137–147.

[2] BAR-YOSSEF, Z., JAYRAM, T. S., KUMAR, R., SIVAKUMAR, D., AND TREVISAN, L. Counting distinct elements in a data stream. In *Randomization and Approximation Techniques (RANDOM)* (2002), J. D. P. Rolim and S. P. Vadhan, Eds., vol. 2483 of *Lecture Notes in Computer Science*, Springer, pp. 1–10. 6th International Workshop, RANDOM 2002, Cambridge, MA, USA, September 13-15, 2002, Proceedings.

[3] BECCHETTI, L., CASTILLO, C., DONATO, D., LEONARDI, S., AND BAEZA-YATES, R. Using rank propagation and probabilistic counting for link-based spam detection. In *Proceedings of the Workshop on Web Mining and Web Usage Analysis (WebKDD)* (2006), ACM Press.

[4] Given an arbitrary partitioning of the original file into subfiles, it suffices to collect register values and apply componentwise a max operation.

[4] BRODER, A. Z. On the resemblance and containment of documents. In *Compression and Complexity of Sequences* (1997), IEEE Computer Society, pp. 21–29.

[5] BRODER, A. Z. Identifying and filtering near-duplicate documents. In *Proceedings of Combinatorial Pattern Matching: 11th Annual Symposium, CPM 2000, Montreal, Canada* (2000), R. Giancarlo and D. Sankoff, Eds., vol. 1848 of *Lecture Notes in Computer Science*, pp. 1–10.

[6] CHASSAING, P., AND GÉRIN, L. Efficient estimation of the cardinality of large data sets. In *Proceedings of the 4th Colloquium on Mathematics and Computer Science, Nancy* (2006), vol. AG of *Discrete Mathematics & Theoretical Computer Science Proceedings*, pp. 419–422. Full paper available at `http://arxiv.org/abs/math.ST/0701347`.

[7] DATAR, M., GIONIS, A., INDYK, P., AND MOTWANI, R. Maintaining stream statistics over sliding windows. *SIAM Journal on Computing 31*, 6 (2002), 1794–1813.

[8] DE BRUIJN, N. G. *Asymptotic Methods in Analysis*. Dover, 1981. A reprint of the third North Holland edition, 1970 (first edition, 1958).

[9] DURAND, M. *Combinatoire analytique et algorithmique des ensembles de données*. PhD thesis, École Polytechnique, France, 2004.

[10] DURAND, M., AND FLAJOLET, P. LOGLOG counting of large cardinalities. In *Annual European Symposium on Algorithms (ESA03)* (2003), G. Di Battista and U. Zwick, Eds., vol. 2832 of *Lecture Notes in Computer Science*, pp. 605–617.

[11] ESTAN, C., VARGHESE, G., AND FISK, M. Bitmap algorithms for counting active flows on high speed links. Technical Report CS2003-0738, UCSD, Mar. 2003. Available electronically. Summary in *ACM SIGCOMM Computer Communication Review* Volume 32 , Issue 3 (July 2002), p. 10.

[12] FLAJOLET, P. On adaptive sampling. *Computing 34* (1990), 391–400.

[13] FLAJOLET, P. Counting by coin tossings. In *Proceedings of ASIAN'04 (Ninth Asian Computing Science Conference)* (2004), M. Maher, Ed., vol. 3321 of *Lecture Notes in Computer Science*, pp. 1–12. (Text of Opening Keynote Address.).

[14] FLAJOLET, P., GOURDON, X., AND DUMAS, P. Mellin transforms and asymptotics: Harmonic sums. *Theoretical Computer Science 144*, 1–2 (June 1995), 3–58.

[15] FLAJOLET, P., AND MARTIN, G. N. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences 31*, 2 (Oct. 1985), 182–209.

[16] GIROIRE, F. Order statistics and estimating cardinalities of massive data sets. In *2005 International Conference on Analysis of Algorithms* (2005), C. Martínez, Ed., vol. AD of *Discrete Mathematics and Theoretical Computer Science Proceedings*, pp. 157–166.

[17] GIROIRE, F. Directions to use probabilistic algorithms for cardinality for DNA analysis. Journées Ouvertes Biologie Informatique Mathématiques (JOBIM'06), 2006.

[18] GIROIRE, F. *Réseaux, algorithmique et analyse combinatoire de grands ensembles*. PhD thesis, Université Paris VI, 2006.

[19] JACQUET, P., AND SZPANKOWSKI, W. Analytical de-Poissonization and its applications. *Theoretical Computer Science 201*, 1-2 (1998), 1–62.

[20] KNUTH, D. E. *The Art of Computer Programming*, 2nd ed., vol. 3: Sorting and Searching. Addison-Wesley, 1998.

[21] KOLCHIN, V. F., SEVASTYANOV, B. A., AND CHISTYAKOV, V. P. *Random Allocations*. John Wiley and Sons, New York, 1978. Translated from the Russian original *Slučajnye Razmeščeniya*.

[22] PALMER, C., GIBBONS, P., AND FALOUTSOS, C. Data mining on large graphs. In *Proceedings of the ACM International Conference on SIGKDD* (2002), pp. 81–90.

[23] SZPANKOWSKI, W. *Average-Case Analysis of Algorithms on Sequences*. John Wiley, New York, 2001.

[24] WHANG, K.-Y., VANDER-ZANDEN, B., AND TAYLOR, H. A linear-time probabilistic counting algorithm for database applications. *ACM Transactions on Database Systems 15*, 2 (1990), 208–229.