Probabilistic Counting of a Large Number of Events – Revisited

Micha Hofri, Nikolaos Kechris

Department of Computer Science, Rice University, Houston, TX 77005-1892
Department of Computer Science, University of Houston, Houston, TX 77204-3475

March 18, 1996

ABSTRACT

We revisit a method suggested in 1978 by R. Morris to obtain randomized, approximate census over large sets in small registers. We demonstrate that his choice of estimation functions lead to a particularly simple method of computing the moments of the estimate. We emphasize statistical aspects of the procedure.

1. Problem Description

The problem of counting a large number of events (items) under storage constraints poses now less technological difficulties than when [6] was published. It is still of theoretical, and sometimes also practical interest. Particular instances may arise in database systems, where accumulating run-time statistics of references and cross-references for a large number of records over long stretches of time can lead to material improvement in the efficiency of query processing; similarly, in certain operating systems the use of such statistics can help improve the scheduling algorithms used and the resulting system performance. We refer the reader to the above reference for a refreshing engineering-oriented view of the problem, and to [1] for a mathematical treatment of more recent vintage.

The nature of the use of these counts implies that precision of the counts of each event type is rarely critical, and we could make do well enough with an approximate measure of the number of occurrences. Since storage is an important factor when a large number of counters have to be maintained concurrently, we assume in the following that we use registers which are too small for the usual representation to be used. Specifically, counting takes place in a b-bit register (which can hold exact counts up to $2^b - 1$). Since the number of occurrences of an event can get fairly large we will investigate methods for counting up to numbers much greater than $2^b - 1$, essentially by recording the logarithm of the (approximate) count, using a suitably selected base. We abide by the constraint that no additional storage is available. This approach falls within the family of *randomized algorithms.*

This is the solution proposed and analyzed by Morris [6]. Flajolet [1] carries the analysis further. The contribution of this report is in showing a different method of analysis that produces exact results for the moments of the estimated count, expressed in closed form, without need for asymptotics. We achieve this by using a slightly different form of the estimate. Asymptotics are needed to address the issue of tail probabilities in a tractable form, and this is done in section 3.

In section 4 we quantify the trade-off between the range of representable counts and the precision of the estimate.

In section 5 we examine an interesting contrast between the "log-based" approach and simple sampling, where at every event, the counter is incremented with a constant probability p. As the results will show, the latter is preferable when we have (even a very rough) *a priori* estimate of the range of likely values of the total number of events. However, if the count actually comes to a small part of the range only, the accuracy of this method can be so poor, that the estimate is meaningless. When an estimate of the range, to within several orders of magnitudes is not available, and in particular if we want to handle concurrently a large number of tallies which may be of very different sizes, the analyzed approach will provide a simple method that guarantees a better relative accuracy.

2. Using the Binary Logarithm

We imagine the register keeps track of the binary logarithm of the number of events that have occurred so far. Specifically, in order for a zero register to represent the state that no event has yet occurred we use the value stored in the register, v, so that[1]

$$v(n) \text{ represents } \lg(1+n), \tag{1}$$

subject to the integrality constraint on v. Our estimate for the number of events that occurred would be then

$$\hat{n}(v) = 2^v - 1. \tag{2}$$

With some abuse of terminology we say that the range of this counting scheme is the largest estimate it can produce. Since the register assumes only integral values, and we want an unbiased estimate, the number of events X_v between the event that caused the counter to achieve the value v till it is next incremented to $v+1$ should have the expected value $2^{v+1} - 2^v = 2^v$. This suggests the use of a probabilistic approach, where we view X_v as a geometrically distributed random variable with a probability of success (leading to an increment) 2^{-v}, so that the expected number of events until an increment occurs is the desired 2^v. This makes the counter content a simple birth process. Using a (pseudo)random number generator on the interval $[0,1]$ we construct the counting algorithm as follows :

```
repeat: {when (new event occurred)
        do r: = random[0,1];
          if r ≤ 1/2^v then v:= v + 1;
        od;}
```

We compute now the distribution of the register content, as a function of the number of events—denoted by n throughout—that has occurred so far, under the assumption that the generator is perfect in terms of uniformity and independence. Let $p_{k,n}$ denote the probability that

[1] $\lg x$ denotes the binary logarithm, $\log_2 x$.

the register contains k after n events occurred. From the algorithm we see,

$$p_{k,n} = (1-2^{-k})p_{k,n-1} + 2^{-(k-1)}p_{k-1,n-1} \qquad k,n \geq 1, \tag{3}$$

and the boundary values $p_{0,n} = \delta_{0,n}$ and $p_{k,0} = \delta_{k,0}$ complete the definition. introduce the probability generating function (*pgf*) $g_n(z) = \Sigma_{k\geq 0} p_{k,n} z^k$. Equation (3) then produces

$$g_n(z) - p_{0,n} = g_{n-1}(z) - p_{0,n-1} - g_{n-1}\left(\tfrac{z}{2}\right) + p_{0,n-1} + z g_{n-1}\left(\tfrac{z}{2}\right), \quad n \geq 1.$$

and we conclude that

$$\begin{aligned} g_0(z) &= 1 \\ g_n(z) &= g_{n-1}(z) + (z-1)g_{n-1}\left(\tfrac{z}{2}\right) \qquad n \geq 1. \end{aligned} \tag{4}$$

There is no difficulty in iterating this recursion to write an explicit solution for $g_n(z)$:

$$g_n(z) = \sum_{j=0}^{n} \binom{n}{j} \prod_{k=0}^{j-1} (z2^{-k} - 1). \tag{5}$$

It is possible to use equation (5) and obtain an explicit expression for the probabilities $p_{k,n}$. Its value for numerical evaluation is doubtful:

$$p_{k,n} = [z^k]g_n(z) = \frac{2^{-k(k-1)/2}}{\prod_{i=1}^{k}(1-2^{-i})} \sum_{j\geq k} \binom{n}{j} (-1)^{j+k} \prod_{l=j-k+1}^{j} (1-2^{-l}). \tag{6}$$

Somewhat simpler-looking expressions for $p_{k,n}$ can be recovered from a different *gf* defined as $q_k(x) = \Sigma_{n\geq 0} p_{k,n} x^n$. Using it with equation (3) we find the recurrence

$$q_k(x) = x(1-2^{-k})q_k(x) + 2^{1-k}xq_{k-1}(x), \quad k \geq 1, \quad q_0(x) = 1. \tag{7}$$

Hence,

$$\begin{aligned} q_k(x) &= \frac{2x2^{-k}}{1-x(1-2^{-k})}\, q_{k-1}(x) = x^k 2^{-\binom{k}{2}} \prod_{j=1}^{k} (1-x(1-2^{-j}))^{-1} \\ &= x^k 2^{-\binom{k}{2}} \sum_{j=1}^{k} \frac{1}{1-x(1-2^{-j})} \prod_{\substack{i=1 \\ i\neq j}}^{k} \frac{2^j-1}{2^{j-i}-1}\,. \end{aligned} \tag{8}$$

The extraction of coefficients leads to

$$q_{k,n} = 2^{-k(k-1)/2} \sum_{j=1}^{k} (1-2^{-j})^{n-1} \prod_{\substack{i=1 \\ i\neq j}}^{k} \frac{2^j}{2^{j-i}-1} = 2^{-\binom{k}{2}} \sum_{j=1}^{k} (1-2^{-j})^{n-1} \Big/ \prod_{\substack{i=1 \\ i\neq j}}^{k} (2^{-i} - 2^{-j}). \tag{9}$$

Remarks: 1. Clearly we should have $p_{k,n} = q_{k,n}$. Since their formulas appear different it is instructive to show they are indeed equal. We do it using a calculation adapted from [3]. The probabilities $q_{k,n}$ are rewritten with the notation

$$(r)_n = \prod_{k=1}^{n} (1-r^k). \tag{10}$$

We only use $r = ½$ here, and its numerical value varies between 1, for $(½)_0$, and 0.2887881... for $(½)_\infty$. Then

$$q_{k,n} = \sum_{j=0}^{k} (1-2^{-j})^n (-1)^{k-j} \frac{2^{-\binom{k-j}{2}}}{(1/2)_j (1/2)_{k-j}}, \tag{11}$$

where a $j=0$ term (which makes no contribution) was added, and a $(1-2^{-j})$ in both the numerator and denominator, to show it is a convolution. Developing the binomial leads to

$$q_{k,n} = \sum_l \binom{n}{l} (-1)^l \sum_{j=0}^{k} [z^j] \sum_{i\geq 0} \frac{2^{-il} z^i}{(1/2)_i} \cdot [z^{k-j}] \sum_{r\geq 0} \frac{(-z)^r 2^{-\binom{r}{2}}}{(1/2)_r}. \tag{12}$$

The convolution, with the Euler identities [2, pp. 89-90]

$$\sum_{i\geq 0} \frac{(uz)^i}{(1-z)(1-z^2)\cdots(1-z^i)} = \prod_{k\geq 1} (1-uz^k)^{-1} \tag{13}$$

and

$$\sum_{r\geq 0} \frac{u^r z^{\binom{r}{2}}}{(1-z)(1-z^2)\cdots(1-z^r)} = \prod_{k\geq 0} (1+uz^k), \tag{14}$$

with some cancellations, provide

$$q_{k,n} = \sum_l \binom{n}{l} (-1)^l [z^k] \prod_{i=0}^{l-1} (1-z2^{-i}).$$

Hence $\Sigma_k z^k q_{k,n}$ is equal to $g_n(z)$ as given in (5). Equivalently, one can show that $\Sigma_n g_n(z) x^n$ and $\Sigma_k q_k(x) z^k$ lead to the same expression.

2. The expressions for $p_{k,n}$ appear different from an analogous one derived in [1], due to our initializing the counter differently (equation (11) is easiest to compare).

3. We shall see that meaningful probabilities are bunched tightly around $k \sim \lg n$, hence computing the sum for $q_{k,n}$ appears vastly more efficient. In fact, though all the expressions we have require summation over terms with alternating signs, the number of contributing terms in (11) is so small—typically, half a dozen—that quite accurate numerical calculations could be done (this is how Table 1 was produced).

4. Our plan to estimate n, the number of events, via its logarithm has an important consequence. It is due to the fact that unbiasedness of an estimator is not preserved under a nonlinear transformation. In particular, an exponential transformation can introduce a sharp distortion (it is easy to manufacture examples with an arbitrary bias). The converse is that it matters little if the counter we use is a poor estimate of $\log n$, so long as the estimate of n constructed from it is unbiased. Tail probabilities, however, can be equally well picked from the distribution of either estimate, as we show below. □

We can use equation (5) further to write an explicit value for the double (Poisson) generating function $V(u,z)$:

$$V(u,z) \equiv e^{-u} \sum_{n\geq 0} g_n(z) \frac{u^n}{n!} = \sum_{i\geq 0} \frac{u^i}{i!} \prod_{k=0}^{i-1} (z2^{-k} - 1) = \sum_{i\geq 0} \prod_{j=1}^{i} \frac{u(z2^{1-j} - 1)}{j}. \tag{15}$$

All our explicit results have no closed forms; it turns out we can obtain what we need for this section in a simpler, more direct way. The key is that the recurrence (4) can be easily solved when z is a nonnegative power of 2. This is all we need in order to calculate the *moments* of our estimate of the number of events, $\hat{n}$. Computing $g_n(2^r)$ provides the rth moment of the estimate.

In particular, for $z = 2$ and $z = 2^2$ we first find:

$$\begin{aligned} g_n(2) &= g_{n-1}(2) + g_{n-1}(1) = g_{n-1}(2) + 1 = \cdots = n+1. \\ g_n(4) &= g_{n-1}(4) + 3g_{n-1}(2) = g_{n-1}(4) + 3n = \cdots = g_0(4) + 3\sum_{k=1}^{n} k = 1 + \frac{3n(n+1)}{2}. \end{aligned} \tag{16}$$

In general we can write from equation (5) (note the upper bound on j, therein lies the great convenience of these evaluations),

$$g_n(2^r) = \sum_{j=0}^{r} \binom{n}{j} \prod_{i=0}^{j-1} (2^{r-i} - 1). \tag{17}$$

Now consider the first and second moments of our estimate $\hat{n}$:

$$\begin{aligned} E[\hat{n}] &= \sum_{k\geq 0} p_{k,n}(2^k - 1) = \sum_{k\geq 0} p_{k,n}2^k - \textstyle\sum_{k\geq 0} p_{k,n} = g_n(2) - 1 = n. \\ E[\hat{n}^2] &= \sum_{k\geq 0} p_{k,n}(2^k - 1)^2 = \textstyle\sum_{k\geq 0} p_{k,n}2^{2k} - 2\sum_{k\geq 0} p_{k,n}2^k + \sum_{k\geq 0} p_{k,n} \\ &= g_n(4) - 2g_n(2) + 1 = \frac{n(3n-1)}{2}. \end{aligned} \tag{18}$$

Thus we obtained,

$$\begin{aligned} E[\hat{n}] &= n. \\ V[\hat{n}] &= \frac{n(n-1)}{2}. \end{aligned} \tag{19}$$

The estimator is unbiased, but its variance is large – the expected error is nearly the size of the estimate itself – which is not surprising, since a deviation of v by one corresponds to doubling (or halving) the estimate. On the other hand, this variance is a poor indicator of deviation probabilities, since except for quite small n, the underlying distribution has very light tails, as we show below.

3. Asymptotic Properties of the Distribution

We consider two issues here. One is the tail probabilities for the estimate $\hat{n}$, and the other is the shape its distribution takes as n increases.

3.1. Tail Probabilities

Tell about saddle point.

3.2. Limiting Distribution

The limiting distribution of k, the register content, became apparent through numerical experimentation. Table 1 illustrates the way the distribution changes with n. It has extremely light tails: the missing values in the table denote probabilities smaller than 10^{-13}.

$(1/2)_\infty = 0.288788095086602421278 9\ldots$

$\boldsymbol{p}\backslash n$	2^4	2^{10}	2^{16}	2^{22}	2^{30}
$p_{1,n}$	0.(4)30517	−−	−−	−−	−−
$p_{2,n}$	0.0266659	−−	−−	−−	−−
$p_{3,n}$	0.3064103	−−	−−	−−	−−
$p_{4,n}$	0.4735007	−−	−−	−−	−−
$p_{5,n}$	0.1736714	−−	−−	−−	−−
$p_{6,n}$	0.0190152	0.(6)33816	−−	−−	−−
$p_{7,n}$	0.0006966	0.0011162	−−	−−	−−
$p_{8,n}$	0.(5)90864	0.0604486	−−	−−	−−
$p_{9,n}$	0.(7)43650	0.3429282	−−	−−	−−
$p_{10,n}$	0.(10)78453	0.4215521	−−	−−	−−
$p_{11,n}$	−−	0.1535465	−−	−−	−−
$p_{12,n}$	−−	0.0194496	0.(6)38883	−−	−−
$p_{13,n}$	−−	0.0009398	0.00116014	−−	−−
$p_{14,n}$	−−	0.(4)18363	0.0610895	−−	−−
$p_{15,n}$	−−	0.(6)15083	0.3433293	−−	−−
$p_{16,n}$	−−	0.(9)53531	0.4207433	−−	−−
$p_{17,n}$	−−	−−	0.1532604	−−	−−
$p_{18,n}$	−−	−−	0.0194547	0.(6)38967	−−
$p_{19,n}$	−−	−−	0.00094361	0.0011608	−−
$p_{20,n}$	−−	−−	0.(4)18532	0.0610995	−−
$p_{21,n}$	−−	−−	0.(6)15320	0.3433355	−−
$p_{22,n}$	−−	−−	0.(9)54790	0.4207306	−−
$p_{23,n}$	−−	−−	−−	0.1532559	−−
$p_{24,n}$	−−	−−	−−	0.0194548	−−
$p_{25,n}$	−−	−−	−−	0.0009437	−−
$p_{26,n}$	−−	−−	−−	0.(4)18534	0.(6)38968
$p_{27,n}$	−−	−−	−−	0.(6)15324	0.0011608
$p_{28,n}$	−−	−−	−−	0.(9)54810	0.0610997
$p_{29,n}$	−−	−−	−−	−−	0.3433356
$p_{30,n}$	−−	−−	−−	−−	0.4207304
$p_{31,n}$	−−	−−	−−	−−	0.1532559
$p_{32,n}$	−−	−−	−−	−−	0.0194548
$p_{33,n}$	−−	−−	−−	−−	0.0009437
$p_{34,n}$	−−	−−	−−	−−	0.(4)18534
$p_{35,n}$	−−	−−	−−	−−	0.(6)15324
$p_{36,n}$	−−	−−	−−	−−	0.(9)54810

Table 1: Selected values of $p_{k,n}$. Missing values $< 10^{-13}$.

4. Generalizing to an Arbitrary Base

The size of the counter, b bits, is typically tailored to the architecture of the machine that performs the counting. Otherwise the arithmetic would be quite expensive. Under certain circumstances the range it provides, defi ned following relation (2), of zero to $2^{2^b-1} - 1$, may prove too small for the envisioned application. One simple solution may be to use a larger counter

(adding a bit nearly squares the range), and doubling the register will usually carry a modest time penalty – but the entire issue arises because storage is at premium. One could then increase the range simply by assuming the content is not a binary logarithm, but a logarithm to some higher base, say c. It will be shown—and a brief reflection will convince the reader—that this change incurs a penalty in the size of the expected error. On the other hand, if the nature of the application is such that the above range is very unlikely to be fully used, we could improve the quality of the estimation by using a base *smaller* than 2.

Hence we are led to generalize our calculation for an arbitrary base c. The specific representation we choose (which differs from the one in [1] and [6] in an inessential way) is such that we imagine the value v stored in the register so that

$$v^{(c)}(n) \text{ represents } \log_c((c-1)n+1), \tag{20}$$

so that our estimate for the number of events that occurred till the counter reached its present value at v would be

$$\hat{n}^{(c)}(v) = \frac{c^v - 1}{c-1}. \tag{21}$$

For the special case $c = 2^{2^d}$ we may see the counter as having an implicit binary point; when d is positive there are d zeroes following the counter, and when it is negative, the rightmost d digits are a binary fraction.

Following the same reasoning as before, if the current state of the register is v we increment the counter at each event with the probability $1/[\hat{n}^{(c)}(v+1) - \hat{n}^{(c)}(v)] = c^{-v}$, to keep our estimate for the number of events unbiased.

Using the same notation as in the previous section we can write,

$$p^{(c)}_{k,n} = (1-c^{-k})p^{(c)}_{k,n-1} + c^{-k+1}p^{(c)}_{k-1,n-1} \qquad k,n \geq 1. \tag{22}$$

Treating equation (22) just as we used (3) provides

$$\begin{aligned} &g^{(c)}_0(z) = 1, \\ &g^{(c)}_n(z) = g^{(c)}_{n-1}(z) + (z-1)g^{(c)}_{n-1}\left(\tfrac{z}{c}\right) \qquad n \geq 1. \end{aligned} \tag{23}$$

We can solve here for the various functions explicitly as above. A glance reveals that they would have precisely the same structure as in section 2, with 'c' replacing the '2' under the product signs. The continuation is similar as well: To find the first two moments of the estimate we have to evaluate $g^{(c)}_n(c)$ and $g^{(c)}_n(c^2)$:

$$g^{(c)}_n(c) = g^{(c)}_{n-1}(c) + (c-1)g^{(c)}_{n-1}(1) = \cdots = g^{(c)}_0(c) + \sum_{k=1}^{n}(c-1) = (c-1)n+1. \tag{24}$$

and

$$\begin{aligned} g^{(c)}_n(c^2) &= g^{(c)}_{n-1}(c^2) + (c^2-1)g^{(c)}_{n-1}(c) = g^{(c)}_{n-1}(c^2) + (c^2-1)(c-1)(n-1) + c^2 - 1 = \cdots \\ &= g^{(c)}_0(c^2) + (c^2-1)(c-1)\sum_{k=1}^{n-1} k + (c^2-1)n = (c^2-1)(c-1)n^2/2 + (c^2-1)(3-c)n/2 + 1. \end{aligned} \tag{25}$$

Then,

$$E[\hat{n}^{(c)}] = \sum_{k\geq 0} p_{k,n}^{(c)} \frac{c^k - 1}{c - 1} = \frac{1}{c-1}\left(\sum_{k\geq 0} p_{k,n}^{(c)} c^k - \sum_{k\geq 0} p_{k,n}^{(c)}\right) = \frac{1}{c-1}\left(g_n^{(c)}(c) - 1\right) = n. \tag{26}$$

and

$$\begin{aligned} E[\left(\hat{n}^{(c)}\right)^2] &= \sum_{k\geq 0} p_{k,n}^{(c)} \left(\frac{c^k - 1}{c - 1}\right)^2 = \frac{1}{(c-1)^2}\left(\sum_{k\geq 0} p_{k,n}^{(c)} c^{2k} - 2\sum_{k\geq 0} p_{k,n}^{(c)} c^k + \sum_{k\geq 0} p_{k,n}^{(c)}\right) \\ &= \frac{1}{(c-1)^2}\left(g_n^{(c)}(c^2) - 2g_n^{(c)}(c) + 1\right) = (c+1)n^2/2 - (c-1)n/2. \end{aligned} \tag{27}$$

Therefore we conclude

$$\begin{aligned} E[\hat{n}^{(c)}] &= n \\ V[\hat{n}^{(c)}] &= \frac{c-1}{2} n(n-1). \end{aligned} \tag{28}$$

The results reduce to the previous ones when we use $c = 2$. Table 2 below presents some numerical examples of the achievable precision when c is of the form 2^{2^d}, for $-5 \leq d \leq 2$.

5. Discussion

We would prefer the reader to view the above as an exercise in the use of generating functions in the analysis of a randomized algorithm, rather than as a recommendation for its use. The presented method is probably not a realistic one to choose for most applications. Storage is not as expensive as when [6] was published and large address spaces are possible. On the other hand, if we consider scenarios where storage would be an issue (where we need to count a large number of parallel streams of fast-coming events, or there would be no need for a large range), the computational load of producing pseudo random variates with a flat distribution at the tail—note that extremely small probabilities may be required here naturally, and most of the efficient random-number generators, which are arithmetic-based, misbehave in such extreme states—may prove far too heavy.

It is interesting to examine here the simple-minded approach of probabilistic counting by sampling with a *fixed* probability p, regardless of the content of the counter. This approach will produce the count ν with $E[\nu|n] = np$, and $V[\nu|n] = np(1-p)$. We compare it to the Morris method when the counter has b bits and the base is given as $c = 2^{2^d}$. This allows for a range of $N = (c^{2^b-1} - 1)/(c-1)$ events. For the two methods to have the same range we want to fix p so that $E[\nu|N] = Np$ equals $2^b - 1$ as well, hence

$$p = \frac{2^b - 1}{N} \approx \frac{2^b(c-1)}{c^{2^b-1}} = (c-1)2^{b-(2^b-1)2^d}. \tag{29}$$

Simple sampling produces the unbiased estimator $\tilde{n} = \nu/p$. Since both estimators are unbiased, and we tailored the parameters for both to provide the same range for the number of events, it is meaningful to compare the methods by the quality of the estimates they provide. Define the relative precision of an unbiased estimator as the ratio of its standard deviation (*sd*) to n.

We use the symbols $\hat{\sigma}$ and $\tilde{\sigma}$ for the *sd*s of $\hat{n}$ and $\tilde{n}$. Clearly $\tilde{\sigma}^2$ is $n(1-p)/p$, and since p is typically very small, we approximate this by n/p. Equations (28) and (29) now say

$$\frac{\hat{\sigma}(n)}{n} \approx \sqrt{(c-1)/2}, \qquad \frac{\tilde{\sigma}(n)}{n} = \frac{1}{\sqrt{np}} \approx \sqrt{\frac{N}{n\lg N}}\,. \tag{30}$$

The precision of simple sampling depends on n, and comparison requires we look at a few possible values of n. If the system designer erred badly, and the range is hardly used, with n coming, say, to $\sqrt{N}$ only, he is punished by a miserable precision:

$$\frac{\tilde{\sigma}(n)}{n}\Big|_{n=\sqrt{N}} \approx \left(\frac{2^{2^{b-1}}}{2^b \lg c}\right)^{1/2} = 2^{(2^{b-1}-b-d)/2}. \tag{31}$$

Using $b = 8$ we get for this relative precision $2^{60-d/2}$ which, for d in the reasonable range, a small integer, is almost meaningless, and clearly insupportable. When n is closer to N, which we represent by $N2^{-r}$, with r a small integer, it looks better:

$$\frac{\tilde{\sigma}(n)}{n}\Big|_{n=N2^{-r}} \approx \left(2^r/(2^b-1)\lg c\right)^{1/2} \approx 2^{(r-b-d)/2}. \tag{32}$$

Some examples are collected in Table 2.

An interesting variation on the topic has been suggested by Kruskal and Greenberg in [5]. They consider the same counting algorithm, but avoid settling on any particular function for the estimate $\hat{n}(v)$ in terms of the counter value. They envision using an arbitrary strictly monotonic function $f(v)$, and setting $p_0^{-1} = 1 + f(0)$, and $p_v^{-1} = f(v) - f(v-1)$ for $v > 0$. This provides a counter evolution such that $E[n|v] = f(v) = \Sigma_{i=0}^{v} 1/p_i - 1$, and a corresponding expression for the variance of n. Note that in the last left-hand side we used n rather than $\hat{n}$: when one uses arbitrary probabilities of incrementation, including such that do not necessarily change exponentially with v in an homogeneous way, the approach we used is infeasible. In particular, there is no relation equivalent to (4), from which to derive the moments of our estimate. In the approach of [5] one must go the Bayesian way, and consider n itself a random variable (possibly with an improper prior).

References.

d	c	$\log_2 N$	$\hat{\sigma}/n$	$\tilde{\sigma}/n, r=2$	$\tilde{\sigma}/n, r=10$
−5	1.0219	13.48	0.105	0.707	11.314
−4	1.0443	20.43	0.149	0.500	8.000
−3	1.0905	35.34	0.213	0.354	5.657
−2	1.1892	66.15	0.308	0.250	4.000
−1	1.4142	128.77	0.455	0.177	2.828
0	2.0000	255.00	0.707	0.125	2.000
1	4.0000	508.42	1.225	0.088	1.414
2	16.0000	1016.09	2.739	0.062	1.000

Table 2
The dependence of the range N and relative precision on the position of the implicit binary point in a counter of 8 bits. $n = N \cdot 2^{-r}$.

[1] P. Flajolet, Approximate Counting: A Detailed Analysis. *BIT,* **25,** 113–134 (1985)
[2] M. Hofri, *Analysis of Algorithms: Mathematical Methods & Computational Tools.* Oxford University Press, New York (1995).
[3] P. Kirschenhofer, H. Prodinger, Approximate Counting: An Alternative Approach. *Inform. Theor. et Applic. (RAIRO),* **25**#1, 43–48 (1991).
[4] P. Kirschenhofer, H. Prodinger, W. Szpankowski, Analysis of a Splitting Process Arising in Probabilistic Counting and Other Related Algorithms. Preprint. (A previous version appeared in W. Kuich (Ed.) *Proc. ICALP* 211–222, Vienna, June 1992.)
[5] J.B. Kruskal, A.G. Greenberg, A Flexible Way of Counting Large Numbers Approximately in Small Registers. *Algorithmica,* **6,** 590–596 (1991).
[6] Robert Morris, Counting Large Number of Events in Small registers. *Comm. of ACM,* **21,** 840–842 (1978).
[7] H. Prodinger, Digital Search Trees and Basic Hypergeometric Functions. *EATCS Bulletin,* **56**, 112–115 (1995).