

ASIAN'04,  
Chiang Mai 2004



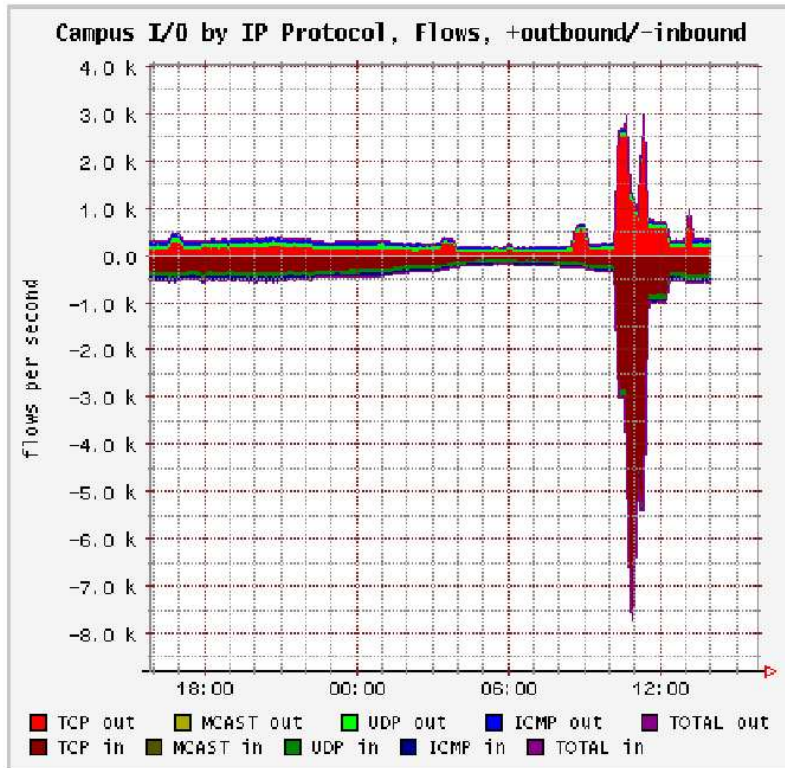
---

# Counting by Coin Tossings

Philippe Flajolet, INRIA, France

<http://algo.inria.fr/flajolet>

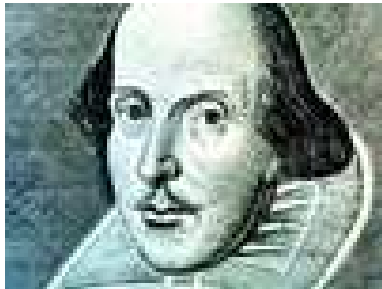
From Estan-Varghese-Fisk: traces of attacks  
Need number of active connections in time slices.



Incoming/Outgoing flows at 40Gbits/second.  
Code Red Worm: 0.5GBytes of compressed data per hour (2001).  
CISCO: in 11 minutes, a worm infected 500,000,000 machines.

The situation is like listening to a play of Shakespeare and at the end *estimate the number of different words*.

Rules: Very little computation per element scanned, very little auxiliary memory.



From Durand-Flajolet, LogLog Counting (ESA-2003):

Whole of Shakespeare,  $m = 256$  small “bytes” of 4 bits each = 128bytes

```
ghffffghfghgghggggghghheehfhfhhgghghghhfgffffhhhiigfhhffgfiihfhhh  
igigighfgihffffghigihghigfhhgeegeghgghhhgghhfhidiigihighihehhffgg  
hfgighigffghdieghhhggghhfgghfiieffghghihifgggffihgihfggighgiiif  
fjgfgjhhjiihfjhgehghfhfhjhiggghghihigghhihihgiighghflgjfgjjjml
```

Estimate  $n^\circ \approx 30,897$  vs  $n = 28,239$  distinct words. Error: +9.4% w/ 128 bytes!

## Uses:

- **Routers:** intrusion, flow monitoring & control
- **Databases:** Query optimization, cf  $\mathfrak{M} \cup \mathfrak{M}'$  for *multisets*; Estimating the size of queries & “sketches”.
- **Statistics gathering:** on the fly, fast and with little memory even on “unclean” data  $\simeq$  layer 0 of “*data mining*”.

## This talk:

- Estimating characteristics of large data streams
  - sampling; size & cardinality & nonuniformity index ( $F_1, F_0, F_2$ )
  - ~> power of randomization via hashing
    - ◇ Gains by a factor of  $>400$  (Palmer *et al.*)
- Analysis of algorithms
  - generating functions, complex asymptotics, Mellin transforms
    - ◇ Nice problems for theoreticians.
- Theory and Practice
  - Interplay of analysis and design ~> super-optimized algorithms.

# 1 **PROB. ALG. ON STREAMS**

Given:  $S$  = a large stream  $S = (r_1, r_2, \dots, r_\ell)$  with duplicates

—  $\|S\|$  = **length** or **size**: total # of records ( $\ell$ )

—  $|S|$  = **cardinality**: # of *distinct* records ( $c$ )

---

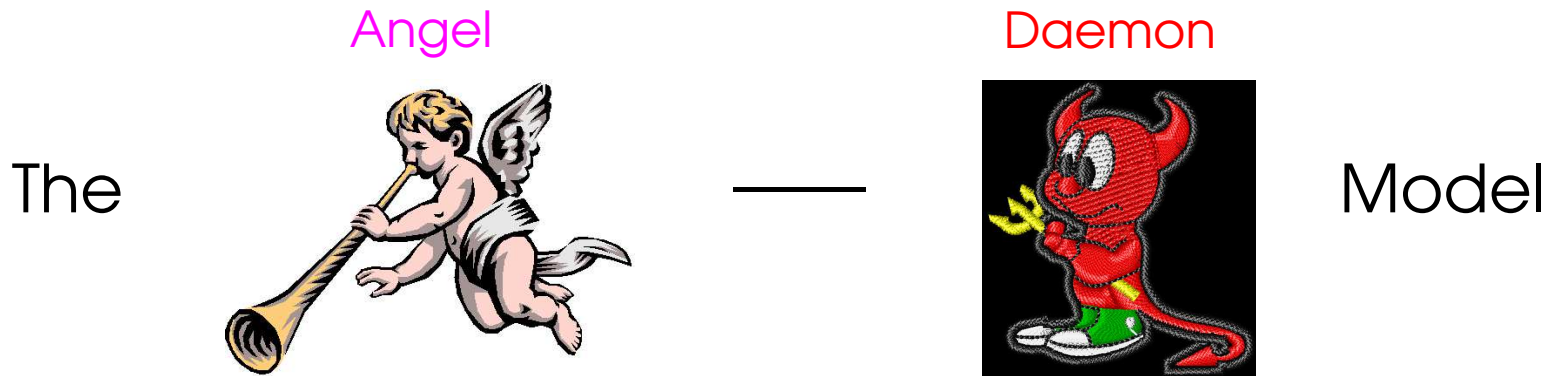
◇ How to estimate size, cardinality, etc?

More generally, if  $f_v$  is frequency of value  $v$ :  $F_p := \sum_{v \in \mathbb{D}} (f_v)^p$ .

Cardinality is  $F_0$ ; size is  $F_1$ ;  $F_2$  is indicator of nonuniformity of distribution;  
“ $F_\infty$ ” is most frequent element (Alon, Matias, Szegedy, STOC96)

◇ How to sample?

— with or without multiplicity



Pragmatic assumptions/ Engineer's point of view:  
 Can get random bits from data: **Works fine!**

(A1) There exists a "good" hash function

$$h : \mathcal{D} \mapsto \mathcal{B} \equiv \{0, 1\}^L$$

Data domain  $\mapsto$  Bits

Typically:  $L = 30-32$  (more or less, maybe).

$$h(x) := \lambda \cdot \langle x \text{ in base } B \rangle \bmod p$$

---

Sometimes, also: (A2) There exists a "good" pseudo-random number gen.  $T : \mathcal{B} \mapsto \mathcal{B}$ , s.t. iterates  $Ty_0, T^{(2)}y_0, T^{(3)}y_0, \dots$  look random. ( $T(y) := (a \cdot y \bmod p)$ )

## Two preparatory examples.

Let a flow of people enter a room.

— *Birthday Paradox*: It takes on average 23 to get a **birthday collision**

— *Coupon Collector*: After 365 persons have entered, expect a **partial collection** of  $\sim 231$  different days in the year; it would take more than 2364 to reach a full collection.

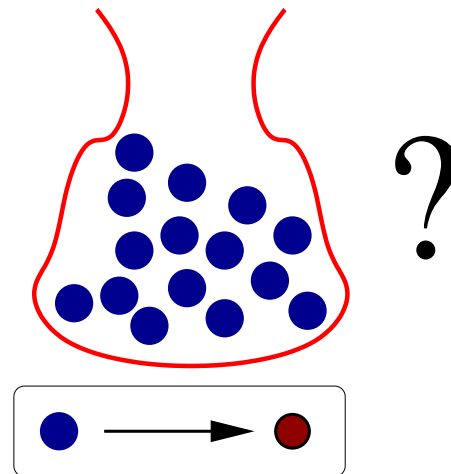
$B$ 1st birthday coll.	$n$	$C$ complete coll.
$\mathbb{E}_n(B) \sim \sqrt{\frac{\pi n}{2}}$	$\approx ne^{-1}$	$\mathbb{E}_n(C) = nH_n \sim n \log n$

Suppose we *didn't know* the number  $N$  of days in the year but could identify people with the same birthday. Could we *estimate*  $N$ ?

## 1.1 Birthday paradox counting

- A warm-up “abstract” example due to Brassard-Bratley (Book 1996) = a *Gedanken* experiment.

How to weigh an urn by shaking it?



Urn contains unknown number  $N$  of balls.

♠ Deterministic: Empty it one by one: cost is  $O(N)$ .

♥ Probabilistic  $O(\sqrt{N})$ : (shake, draw, paint)\*; stop!

---

ALG: Birthday Paradox Counting

Shake, pull out a ball, mark it with paint;

repeat until draw an already marked ball.

Infer  $N$  from  $T$  = number of steps.

---

We have  $\mathbb{E}(T) \sim \sqrt{\pi N/2}$  by Birthday Paradox.

• Invert and try  $X := \frac{2}{\pi}T^2$ . Estimate is biased

•• Analyse 2nd moment of BP, find  $\mathbb{E}(T^2) \sim 2N$  and propose  $X := T^2/2$ . Estimate is now (asymptotically) unbiased.

••• Wonder about accuracy: Standard Error :=  $\frac{\text{Std Deviation of estimate } (X)}{\text{Exact value } (N)}$ .

$\leadsto$  Need to analyse fourth moment  $\mathbb{E}(T^4)$ . Do maths:

$$\mathbb{E}_N(T^{2r}) = 2^r r! N^r, \quad \mathbb{E}_N(T^{2r+1}) = (1 \cdot 3 \cdots (2r-1)) \sqrt{\frac{\pi}{2}} N^{r+\frac{1}{2}}.$$

$\implies E(T^4) \sim 8N^2$ . Standard error  $\implies$  Estimate  $\in (0, 3N)$ . ( $N = 10^6$ ): 384k; 3,187k; 635k; 29k; 2,678k; 796k; 981k, ...

•••• Improve algorithm. Repeat  $m$  times and average.

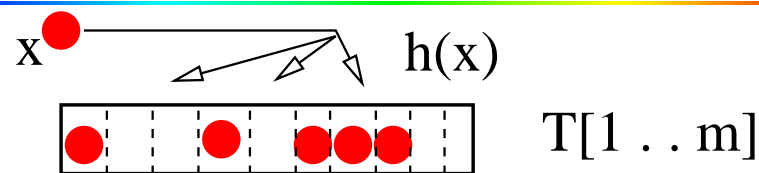
$\leadsto$  Time cost:  $O(m\sqrt{N})$  for accuracy  $O\left(\frac{1}{\sqrt{m}}\right)$ .

Shows usefulness of maths: Ramanujan's  $Q(n)$  function, Laplace's method for sums or integrals (cf Knuth, Vol 1); singularity analysis. . .

## 1.2 Coupon Collector Counting

First Counting Algorithm: Estimate cardinalities  $\equiv$  # of **distinct** elements.

This is real CS, motivated by *query optimization in data bases*. (Whang et al, ACM TODS 1990)



ALG: Coupon Collector Counting

Given multiset  $S = (s_1, \dots, s_\ell)$ ; Estimate  $\text{card}(S)$ ?

Set up a table  $T[1 \dots m]$  of  $m$  bit-cells.

— for  $x$  in  $S$  do mark cell  $T[h(x)]$ ;

Return  $-m \log V$ , where  $V :=$  fraction of empty cells.

Simulate hashing table; Alg. is indep. of replications.

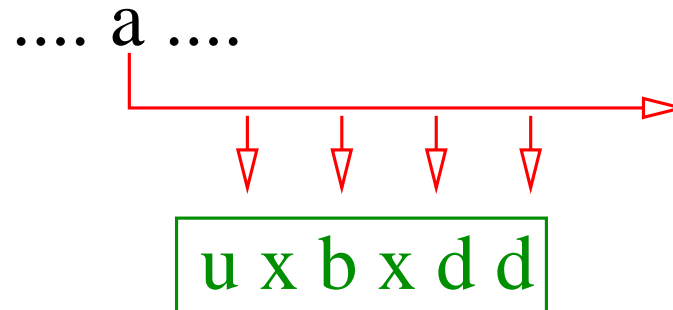
Let  $n$  be sought cardinality. Then  $\alpha := n/m$  is *filling ratio*. Expect  $V \approx e^{-\alpha}$  empty cells by classical analysis of occupancy. Distribution is concentrated. Invert!

Count cardinalities till  $N_{\max}$  using  $\frac{1}{10} N_{\max}$  bits, for accuracy (standard error) = 2%.

Generating functions for occupancy; Stirling numbers; basic depositions.

## 2 SAMPLING

A very classical problem (Vitter, ACM TOMS 1985)



ALG: Reservoir Sampling (*with multiplicities*)

Sample  $m$  elements from  $S = (s_1, \dots, s_N)$ ; ( $N$  unknown a priori)

Maintain a cache (reservoir) of size  $m$ ;

— for each coming  $s_{t+1}$ :

place it in cache with probability  $m/(t+1)$ ; drop random element;

Math: Need analysis of skipping probabilities.

Complexity of Vitter's best alg. is  $O(m \log N)$ .

Useful for building "sketches", order-preserving H-fns & DS.

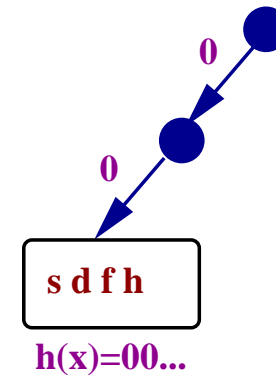
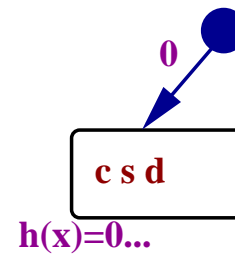
Can we sample *values* (i.e., without multiplicity)?

Algorithm due to (Wegman, ca 1984, unpub.), analysed by (F.1990).

---

Sample of size  $\leq b$ :  
depth  $d = 0, 1, 2, \dots$

**c x a s d**



ALG: Adaptive Sampling (*without multiplicities*)

Get a sample of size  $m$  from  $S$ 's values.

Set  $b := 4m$  (bucket capacity);

— Oversample by adaptive method;

– Get sample of  $m$  elements from the ( $b \equiv 4m$ ) bucket.

---

## Analysis.

View collection of records as a set of bitstrings.

Digital tree aka trie, paged version:

$$\left[ \begin{array}{l} \text{Trie}(\omega) \equiv \omega \text{ if } \text{card}(\omega) \leq b \\ \text{Trie}(\omega) = \overbrace{\text{Trie}(\omega \setminus 0) \quad \text{Trie}(\omega \setminus 1)}^{\bullet} \text{ if } \text{card}(\omega) > b \end{array} \right.$$

(Underlies dynamic and extendible hashing, paged DS, etc)

Refs: (Knuth Vol 3), (Sedgewick, Algorithms), Books by Mahmoud, Szpankowski. General analysis by (Clément-F-Vallée, Alg. 2001), etc.

Depth in Adaptive Sampling is length of leftmost branch;  
Bucket size is # of elements in leftmost page.

For recursively defined parameters:  $\alpha[\omega] = \beta[\omega \setminus 0]$ :

$$\mathbb{E}_n(\alpha) := \sum_{k=0}^n \frac{1}{2^n} \binom{n}{k} \mathbb{E}_k(\beta).$$

Introduce *exponential generating functions (EGF)*:

$A(z) := \sum_n \mathbb{E}_n(\alpha) \frac{z^n}{n!}$  &c. Then  $A(z) = e^{z/2} B\left(\frac{z}{2}\right)$ .

For recursive parameter  $\phi$ :  $\Phi(z) = e^{z/2} \Phi\left(\frac{z}{2}\right) + \text{Init}(z)$

Solve by iteration, extract coefficients; Mellin-ize  $\rightsquigarrow$  later!

**Bonus:** Second Counting Algorithm for *cardinalities*.

Let  $d :=$  sampling *depth*;  $\xi :=$  sample size.

**Theorem [F90]:**  $X := 2^d \xi$  estimates the cardinality of  $S$  using  $b$  words of memory, in a way that is unbiased and with standard error  $\approx 1.20/\sqrt{b}$ .

- $1.20 \doteq 1/\sqrt{\log 2}$ : with  $b = 1,000W$ , get 4% accuracy.
- Distributional analysis by (Louchard RSA 1997).
- Related to folk algorithm for **leader election** on channel: “*Talk, flip coin if noisy; sleep if Tails; repeat!*”
- Related to “**tree protocols with counting**”  
» Ethernet. Cf (Greenberg-F-Ladner JACM 1987).

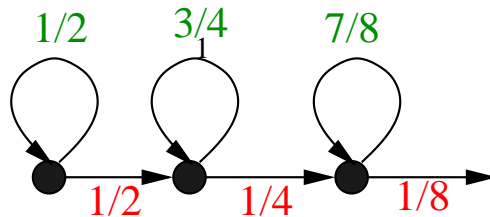
### 3 APPROXIMATE COUNTING

The oldest algorithm (Morris CACM:1977), analysis (F, 1985).

Maintain  $F_1$ , i.e., counter subject to  $C := C + 1$ .

**Theorem:** Count till  $n$  probabilistically using  $\log_2 \log n + \delta$  bits, with accuracy about  $0.59 \cdot 2^{-\delta/2}$ .

Beats information theory(!?): 8 bits for counts  $\leq 2^{16}$  w/ accuracy  $\approx 15\%$ .



ALG: Approximate Counting

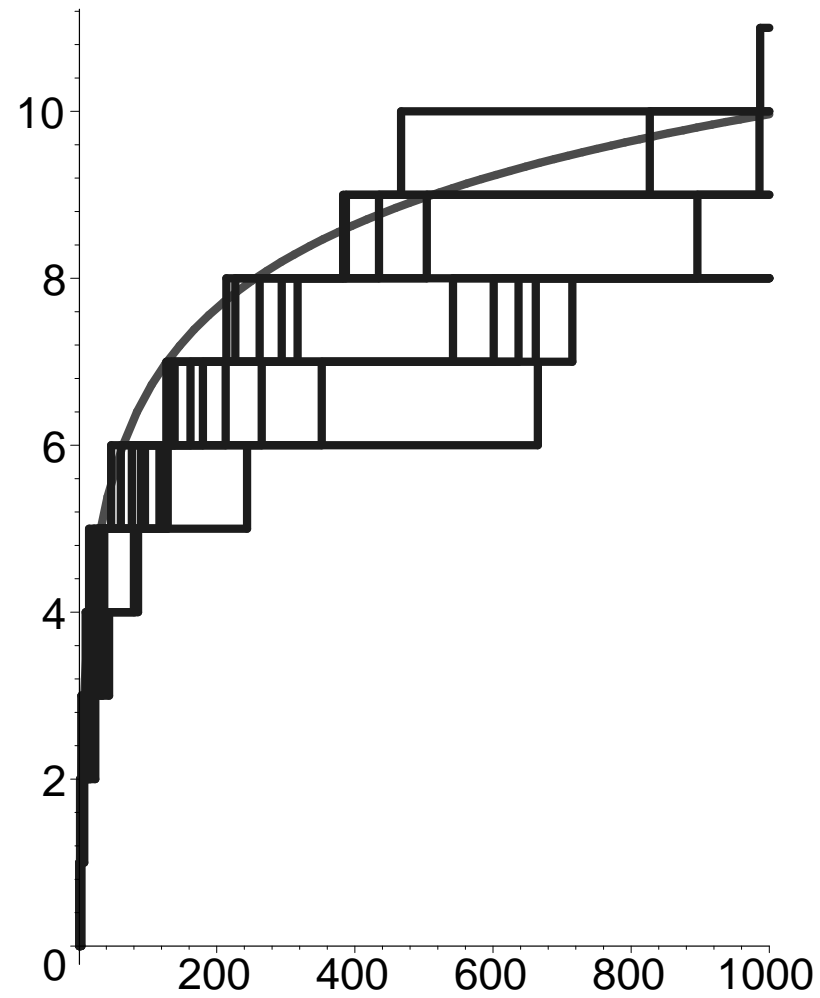
Initialize:  $X := 1$ ;

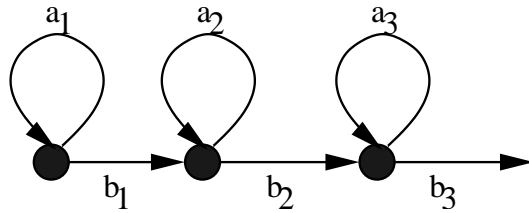
Increment: do  $X := X + 1$  with probability  $2^{-X}$ ;

Output:  $2^X - 2$ .

In base  $q < 1$ : increment with probability  $q^X$ ; output  $(q^{-x} - q^{-1}) / (q^{-1} - 1)$ ; use  $q = 2^{-2^{-\delta}} \approx 1$ .

10 runs of of APCO: value of  $X$  ( $n = 10^3$ )





## Methodology:

♡ Paths in graphs  $\mapsto$  Generating Functions:

$$(f_n) \mapsto f(z) := \sum_n f_n z^n.$$

Here: Symbolically describe all paths:

$$\frac{(a_1)^* b_1 (a_2)^* b_2 (a_3)^*}{\frac{1}{1-a_1} b_1 \frac{1}{1-a_2} b_2 \frac{1}{1-a_3}} \quad \text{since} \quad \frac{1}{1-f} = 1 + f + f^2 + \dots \simeq (f)^*.$$

Perform probabilistic valuation  $a_j \mapsto q^j$ ;  $b_j \mapsto 1 - q^j$ :

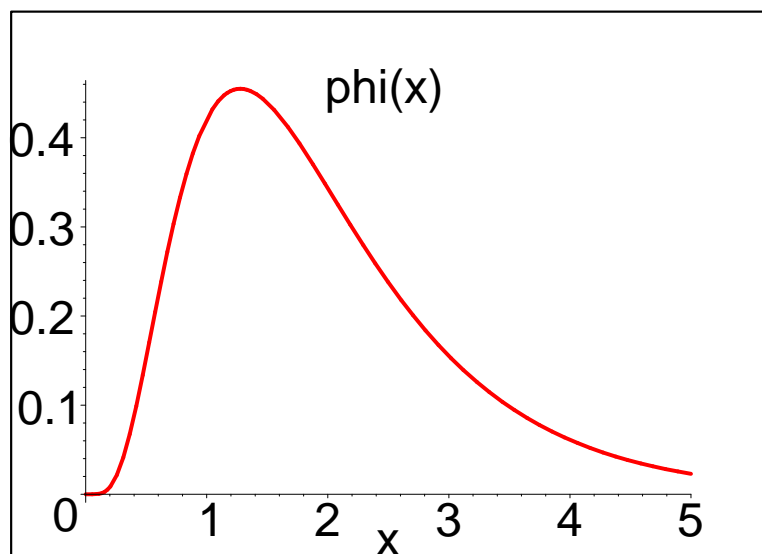
$$H_3(z) = \frac{q^{1+2} z^2}{(1 - (1 - q)z)(1 - (1 - q^2)z)(1 - (1 - q^3)z)}.$$

♡ (Prodinger'94) Euler transform  $\xi := z/(1-z)$ :  $zH_k(z) = \frac{q^{\binom{k}{2}} \xi^{k-1}}{(1 - \xi q) \cdots (1 - \xi q^k)}.$

Exact moments of  $X$  and estimate  $q^X$  via Heine's transformation of  $q$ -calculus:  
 mean is unbiased, variance  $\rightsquigarrow 0.59$ .

♥ Partial fraction expansions  $\leadsto$  asymptotic distribution  
 = quantify typical behaviour + risk! (Exponential tails  $\gg$  Chebyshev ineq.)

We have  $\mathbb{P}_n(X = \ell) \sim \phi(n/2^\ell)$ , where  $((q)_j := (1 - q) \cdots (1 - q_j).$ )



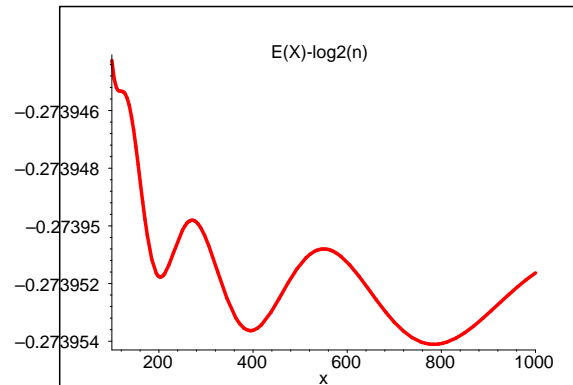
$$\phi(x) := \sum_{j \geq 0} \frac{(-1)^j q \binom{j}{2} e^{-xq^{-j}}}{(q)_\infty (q)_j}$$

♣ Fluctuations:  $\dots, \frac{n}{2^L}, \dots, \frac{n}{4}, \dots$  depend on  $L = \lfloor \log_2 n \rfloor$ .  
 cf. Szpankowski, Mahmoud, Fill, Prodinger, ...

Analyse storage utilization via Mellin transform

## Approximate Counting

Mean( $X$ )  $- \log_2 n$ :



The Mellin transform (F. Régnier Sedgewick 1985); (FIGoDu 1995)

$$f^*(s) := \int_0^{\infty} f(x)x^{s-1} dx.$$

( $P_1$ ) Mapping properties (complex analysis):

Asympt( $f$ )  $\longleftrightarrow$  Singularities( $f^*$ ). Pole at  $\sigma \longrightarrow \boxed{\approx x^\sigma}$

( $P_2$ ) Harmonic sums (superposition of models)

$$\left[ \sum_k \lambda_k f(\mu_k x) \right] \mapsto \left[ \sum_k \lambda_k (\mu_k)^{-s} \right] f^*(s).$$

---

♣ EXAMPLE: dyadic sum,  $F(x) = \sum \phi\left(\frac{x}{2^\ell}\right) \rightsquigarrow F^*(s) = \frac{f^*(s)}{1 - 2^s}$ .

Standard asymptotic terms +  $x^{i\chi} \equiv \exp(i\chi \log x)$ .

## Cultural flashes

- Morris (1977): Counting a large number of events in small memory.
- The power of probabilistic machines & approximation (Freivalds IFIP 1977)
- The FTP protocol: Additive Increase Multiplicative Decrease (AIMD) leads to similar functions (Robert et al, 2001)
- Probability theory: Exponentials of Poisson processes (Yor et al, 2001)

## 4 **CARDINALITY ESTIMATORS**

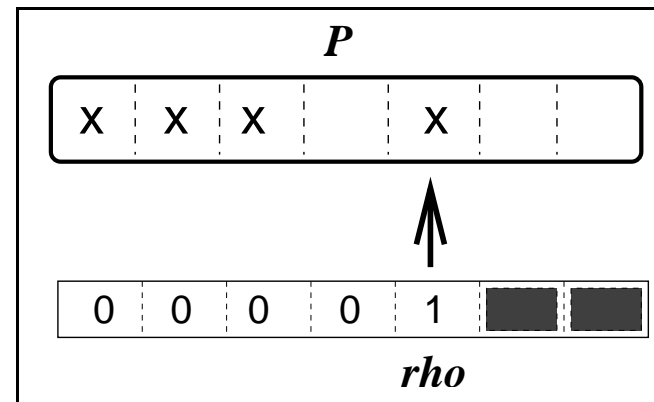
$F_0$  = Number of different values

- 1983–1985: (F-Martin, FOCS+JCSS) Probabilistic Count.
- 1987–1990: (Whang et al) Coupon Coll. Counting
- 1984–1990: (Wegner) (F90 COMP) Adaptive Sampling
- 1996: (Alon et al, STOC)  $F_p$  statistics  $\rightsquigarrow$  later
- 2000: (Indyk FOCS) Stable Law Counting  $\rightsquigarrow$  later
- 2001: (Estan-Varghese SIGCOMM) Multiresolution Bitmap
- 2003: (Durand-F ESA) Loglog Counting

## 4.1 Probabilistic Counting

Third Counting Algorithm  
for cardinalities

:



ALG: Probabilistic Counting

Input: a stream  $S$ ; Output: cardinality  $|S|$

For each  $x \in S$  do /\*  $\rho \equiv$  position of leftmost 1-bit \*/

Set  $\text{BITMAP}[\rho(\text{hash}(w))] := 1$ ; od;

Return  $P$  where  $P$  is position of first 0.

- $P$  estimates  $\log_2(\varphi n)$  for  $\varphi \doteq 0.77351$
- Average over  $m$  trials  $A = \frac{1}{m}[A_1 + \cdots + A_m]$ ; return  $\frac{1}{\varphi} 2^A$ .
- In fact, use stochastic averaging, which needs only *one* hash function:  $S \mapsto (S_{000}, \dots, S_{111})$ .
- Analysis provides

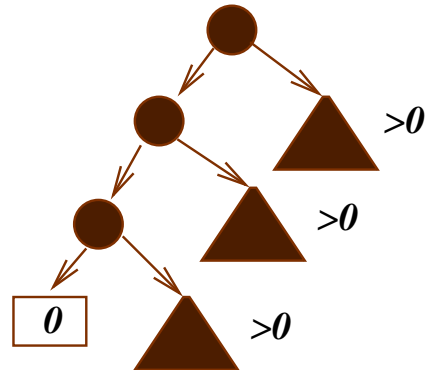
$$\varphi = \frac{e^\gamma}{\sqrt{2}} \prod_{m \geq 2}^* m^{\epsilon(m)}, \quad \epsilon(m) := (-1)^{\sum \text{bits}(m)}.$$

$\epsilon(19) = \epsilon(\langle 10011 \rangle_2) = (-1)^3 = -1$ . Standard error is  $0.78/\sqrt{m}$  for  $m$  Words of  $\log_2 N$  bits. + Exponential Tails  $\gg$  Chebyshev.

(AMS96) and subsequent literature claim wrongly that *several* hash functions are needed!

**Theorem [FM85]:** Prob. Count. is asymptotically unbiased. Accuracy is  $\frac{0.78}{\sqrt{m}}$  for  $m$  Words of size  $\log_2 N$ . E.g. 1,000W = 4kbytes  $\sim$  2.5% accuracy.

Proof:  
trie analysis



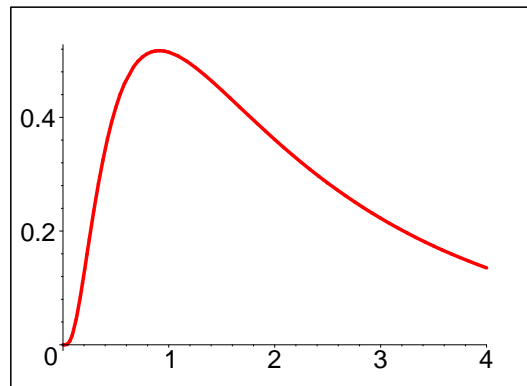
$$1 \cdot (e^{x/8} - 1)(e^{x/4} - 1)(e^{x/2} - 1)$$

$$(1 - q)(1 - q^2)(1 - q^4) \cdots = \sum_n \overbrace{(-1)^{\sum \text{bits}(n)}}^{\epsilon(n)} q^n.$$

Distribution:

$$Q(x) := e^{-x/2} \prod_{j=0}^{\infty} (1 - e^{-x2^j})$$

$$\mathbb{P}_n(X = \ell) \sim Q\left(\frac{n}{2^\ell}\right)$$



+ Mellin requires  $N(s) := \sum_{n \geq 1} \frac{\epsilon(n)}{n^s}$ . One finds  $\log_2 \varphi \equiv -\Gamma'(1) - N'(0) + \frac{1}{2}$ , &c.

## Data mining of the Internet graph

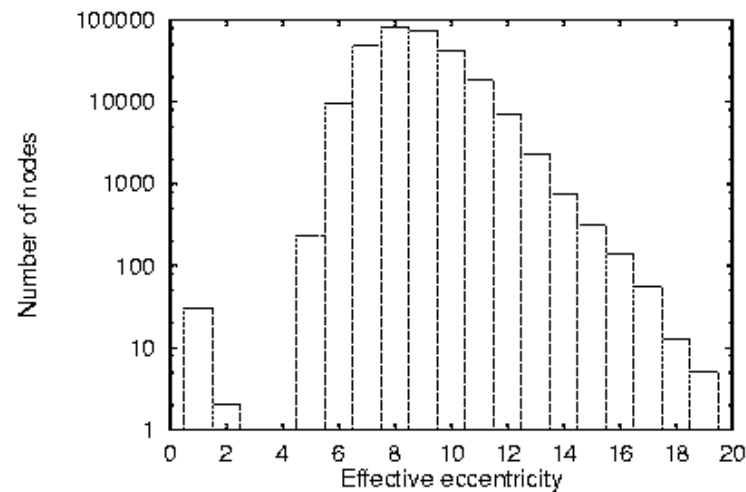
(Palmer, Gibbons, Faloutsos<sup>2</sup>, Siganos 2001)

Internet graph: 285k nodes, 430k edges.

For each vertex  $v$ , define ball  $B(v; R)$  of radius  $R$ .

Want: histograms of  $|B(v, R)|$   $R = 1 \dots 20$

Get it in minutes of CPU rather than a day (400× speedup)



b) Histogram of diameters

Update procedure:  $(h - 1) \mapsto h$  is

for each edge  $(u, v)$  do  $B(v, h) := B(v, h) \cup B(u, h - 1)$

Use: *Probabilistic Counting*. Operate *in core*.

$$\text{Card}_{PC}(\mathcal{S} \cup \mathcal{T}) = \text{Card}_{PC}(\mathcal{S}) \vee \text{Card}_{PC}(\mathcal{T}).$$

where  $\text{Card}_{PC}$  is BITMAP evaluator of cardinalities.

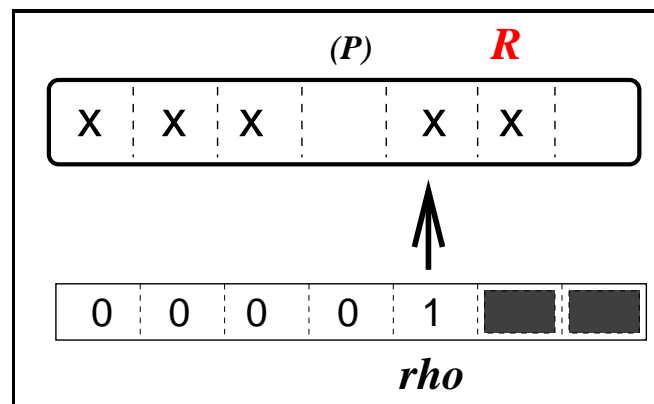
Allows for *fully distributed implementation*.

## 4.2 LogLog Counting

Fourth Counting Algorithm for cardinalities: (Durand-F, ESA 2003)

**Claim: the best algorithm on the market!**

- Hash values and get  $\rho(h(x)) =$  position of leftmost 1-bit = a geometric RV  $G(x)$ .
- To set  $S$  associate  $R(S) := \max_{v \in S} G(v)$ .



- Max of geometric RVs are well-known (Prodinger\*).  
 $R(s)$  estimates  $\sim \log(\hat{\varphi} \text{card}(S))$ , with  $\hat{\varphi} := e^{-\gamma} \sqrt{2}$ .

- Do **stochastic averaging** with  $m = 2^\ell$ :

E.g.,  $S \cong \langle S_{00}, S_{01}, S_{10}, S_{11} \rangle$ : count separately.

Return  $\frac{m}{\hat{\varphi}} 2^{\text{Average}}$ .

- ++ Switch to **Coupon Collector Counting** for small cardinalities.
- ++ Optimize by pruning discrepant values  $\rightsquigarrow$  **superLogLog**.

**Theorem.** LogLog needs  $m$  “bytes”, each of length  $\log_2 \log N$ .  
 Accuracy is:  $\frac{1.30}{\sqrt{m}}$  (BASIC) or  $\frac{0.95}{\sqrt{m}}$  (SUPER)

PROOF: Generating Functions + Saddle-point depoissonization  
 (Jacquet-Szpankowski) + Mellin.  $1.30 \doteq \sqrt{\frac{1}{12} \log^2 2 + \frac{1}{6} \pi^2}$ .



Whole of Shakespeare:

$m = 256$  small “bytes” of 4 bits each = 128bytes

```
ghfffghfghgghggggghghheehfhhfhggghghghhfgffffhhhiigfhhffgfiihfhhh
igigighfgihffffghigihghigfhhgeegeghgghhhgghhfhidiigihighihehhhfgg
hfgighigffghdieghhhggghhfgghfiiheffghghihifgggffihgihfggighgiif
fjgfgjhhjiihfjhgehgghfhhfhjhiggghghihigghhihihgiihgfhlgjfgjjmfl
```

Estimate  $n^\circ \approx 30,897$  against  $n = 28,239$  distinct words

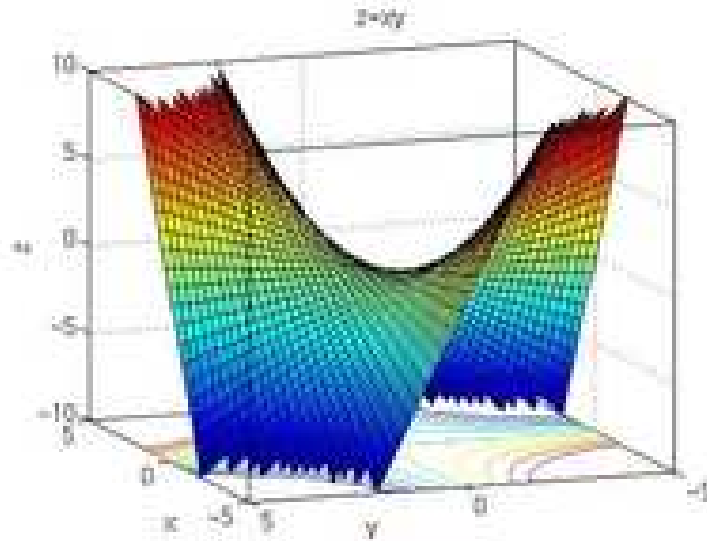
Error is +9.4% for 128 bytes(!!)

An aside: **Analytic depoissonization** (JaSz95<sup>+</sup>)

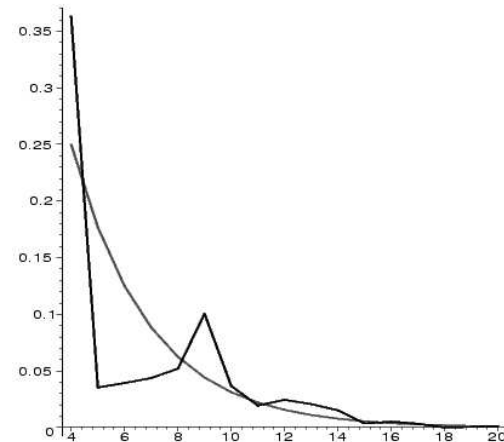
• PROBLEM: Recover asympt.  $f_n$  from  $f(z) = \sum_n f_n \frac{z^n}{n!}$ ?

• Intuition: “with luck”  $f_n \sim \phi(n)$  where  $\phi(z) := e^{-z} f(z)$  is Poisson g.f..  
(Here: “Luck” means good lifting of  $\phi(z)$  to  $\mathbb{C} \equiv$  Poisson flow of complex rate!)

$$f_n = \frac{n!}{2i\pi} \oint f(z) \frac{dz}{z^{n+1}} \approx \phi(n)$$



**Features:** Errors  $\approx$  *Gaussian*, seldom more than  $2\times$  standard error.  
 Algorithm *scales down* (for small cardinalities) and Algorithm *scales up*  
 (large memory size): **HYBRIDIZE with Collision Counting**.



**Mahābhārata:** 8MB, 1M words, 177601 diff.

**HTTP server:** 400Mb log pages 1.8 M distinct req.

$m$	$2^6$ (50by)	$2^{10}$ (0.8kby)	$2^{14}$ (12kb)	$2^{18}$ (200kb)
Obs:	8.9%	2.6%	1.2%	0.32%
$\sigma$ :	11%	2.8%	0.7%	0.36%

## Summary

Analytic results ( $\lg \equiv \log_2$ ): Alg/Mem/Accuracy

CouponCC	AdSamp	ProbC	LogLog
$\approx \frac{N}{10}$ bits	$m \cdot \lg N$ Words	$m \cdot \lg \frac{N}{m}$ Words	$m \cdot \lg \lg \frac{N}{m}$ Bytes
$\approx 2\%$	$\frac{1.20}{\sqrt{m}}$ W	$\frac{0.78}{\sqrt{m}}$ W	$\approx \frac{1.30-0.95}{\sqrt{m}}$ By

**$F_0$  statistics,  $N = 10^8$  & 2% error**

- Coupon Collector Counting = 1 Mbyte
- Adaptive Sampling = 16 kbytes
- Probabilistic Counting: = 8 kbytes
- Multiresolution bitmap (analysis?) = 5 kbytes?
- Loglog Counting = 2 kbytes

(NB: LogLog counting + compression  $\rightsquigarrow \lg \lg N + O(m)$  bits !?)

## 5 FREQUENCY MOMENTS

### 5.1 AMS's $F_2$ algorithm

Recall: Alon, Matias, Szegedy (STOC 1996)<sup>\*\*\*</sup>

$$F_2 := \sum_v (f_v)^2,$$

where  $f_v$  is frequency of value  $v$ .

A beautifully simple idea:  $\text{flip}(x) \equiv \epsilon(x) = \pm 1$  based on  $\text{hash}(x)$ .

---

ALG: F2;

Initialize  $Z:=0$ ;

For each  $x$  in  $S$  do  $Z := Z + \text{flip}(x)$ .

Return  $Z^2$ .

Collect  $m$   $Z$ -values and average, with  $T$ -transform.

---

$$\mathbb{E}(Z^2) = \mathbb{E} \left( \sum_{x \in S} \epsilon(v) \right)^2 = \mathbb{E} \left( \sum_j f_j \cdot \epsilon(j) \right)^2 = \sum_j (f_j)^2.$$

(Actually, they prove stronger complexity result by complicated (impractical?) algorithm.) (What about stochastic averaging?)

## 5.2 Indyk's $F_p$ algorithm

A beautiful idea of Piotr Indyk (FOCS 2000)<sup>\*\*\*</sup> for  $F_p, p \in (0, 2)$ .

- Stable law of parameter  $p \in (0, 2)$ :  $\mathbb{E}(e^{itX}) = e^{-|t|^p}$ .

No second moment; no 1st moment if  $p \in (0, 1)$ .

$$c_1 X_1 + c_2 X_2 \stackrel{\mathcal{L}}{\cong} \mu X, \text{ with } \mu := (c_1^p + c_2^p)^{1/p}.$$

---

ALG:  $F_p$ ;

Initialize  $Z:=0$ ;

For each  $x$  in  $S$  do  $Z := Z + \text{Stable}_\alpha(x)$ .

Return  $Z$ .

Estimate  $F_p$  parameter from  $m$  copies of  $Z$ -values.

---

Remark: Use of  $\log(|Z|)$  to estimate seems better than median(?)

## 6 CONCLUSIONS

For streams, using **practically  $O(1)$  storage**, one can:

- **Sample** positions and even distinct values;
- **Estimate  $F_0, F_1, F_2, F_p$**  ( $0 < p \leq 2$ ) even for *huge* data sets;
- Need **no assumption on nature of data**.



The algorithms are based on **randomization**  $\mapsto$  **Analysis fully applies**

- They **work exactly** as predicted on real-life data;
- They often have **a wonderfully elegant structure**;
- Their analysis involves **beautiful methods** for AofA: “Symbolic modelling by generating functions, Singularity analysis, Saddle Point and analytic depositions, Mellin transforms, stable laws and Mittag-Leffler functions, etc.”

*That's All, Folks!*

