Probabilistic Counting: 
from analysis

to algorithms to programs

Philippe Flajolet, INRIA, Rocquencourt

http://algo.inria.fr/flajolet
Give a (large) sequence $s$ over some (large) domain $D$,

$$s = s_1 s_2 \cdots s_\ell, \quad s_j \in D,$$

View sequence $s$ as a multiset

$$M = m_1^{f_1} m_2^{f_2} \cdots m_n^{f_n}.$$

— A. **Length** := $\ell$;
— B. **Cardinality** := $\text{card}\{s_j\} \equiv n$;
— C. **Mice** := # elements repeated $1, 2, \ldots, 10$ times;
— D. **Icebergs** := # elem. with relative frequency $\frac{1}{\ell} f_v > \frac{1}{100}$;
— E. **Elephants** := # elem. with absolute frequency $f_v > 200$;
— F. **Frequency moments** := $(\sum f_v^r)^{1/r}$.

Alon, Matias, Szegedy; Bar-Yossef; Indyk; Motwani; RAP@Inria . . .
Fl-Martin (1985); Fl (1992); Louchard (1997); Durand-Fl (2003); FlFuGaMe $\rightsquigarrow$ AofA07,
Prodinger, Fill-Janson-Mahmoud-Szpankowski . . .
\[ s = s_1 s_2 \cdots s_\ell, \quad s_j \in D. \]

Length can be \( \ell \gg 10^9 \). Cardinality can be \( n \propto 10^7 \).

Routers in the range of Terabits/sec \((10^{12} \text{ b/s})\).

Google indexes 6 billion pages & prepares to index 100 Petabytes of data \((10^{17} \text{ B})\).

Can estimate a few key characteristics, \textit{QUICK} and \textit{EASY}.
Rules of the game

- **Limited storage**: cannot store elements; use \( \approx \text{one page} \) of print \( \equiv 4\text{kB} \).
- **Limited time**: proceed online = single pass, read once data.
- Allow to **estimate** rather than compute exactly.

Assume **hash function** \( h : \mathcal{D} \mapsto [0, 1] \) scrambles data *uniformly*:

*Angel-daemon scenario*: \( n \) values, replicated and permuted at will, then made into *random uniform* \([0, 1]\).
What for?
— Network management, worms and viruses, traffic monitoring
— Databases: Query optimization = size estimation; also “sketches”.
— Document classification (Broder), cf Google, citeseer, …
— Data mining of web graph, internet graph, etc

Traces of attacks: Number of active connections in time slices.

(Raw ADSL traffic)
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.
Left: ADSL FT@Lyon $1.5 \times 10^8$ packets (21h–23h). Right: (Estan-Varghese-Fisk) different incoming/outgoing connections
Claims:
— High Tech algorithms based on probabilities.
— Efficient programs: Produce short algorithms & programs with $O(10)$ instructions. Gains by factors in the range 100-1000 (!)
— No maths, no algorithms!

AofA: Symbolic methods and generating functions, complex asymptotics (singularities, saddle-point), limit laws and quasipowers, transforms (Mellin), analytic depoisssonization. . .

Constants play a crucial rôle.
1 APPROXIMATE COUNTING

In streaming framework: given $s_1 s_2 \cdots s_\ell$, get length $\ell$.
Means: maintain an efficient counter of events.

The oldest algorithm (Morris CACM:1977): Counting a large number of events in small memory.

Approximate Counting

- Information Theory: need \( \log_2 N \) bits to count till \( N \).
- Approximate counting: use \( \log_2 \log N + O(1) \) for \( \varepsilon \)-approximation, in relative terms and in probability.

How to find an unbounded integer while posing few questions?
- Ask if in \((1-2), (2-4), (4-8), (8-16)\), etc?
- Conclude by binary search (cost is \( 2 \log_2 n \)).

= A general paradigm for unbounded search:

- **Ethernet** proceeds by period doubling + randomization.
- Wake up procedures for **mobile communication** (Lavault+).
- **Adaptive data structures**: e.g., extendible hashing tables.

♥ Approximate Counting
Emulate a counter subject to $X := X + 1$.

**Algorithm: Approximate Counting /* binary base */**

- Initialize: $C := 1$;
- Increment: do $C := C + 1$ with probability $2^{-C}$;
- Output: $2^C - 2$.

Alternate base $q \rightarrow 1$ controls cost/accuracy tradeoff.
Expect $C$ near $\log_2 n$ after $n$ steps, then use only $\log_2 \log n$ bits.

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

**Theorem:**
- Basic binary algorithm is unbiased: $E_n(2^C - 2) = n$.
- Accuracy, i.e., standard error $\equiv \frac{\text{std-dev.}}{n}$ is $\sim \frac{1}{\sqrt{n}}$.
- Asymptotics of distribution is (binary case):

$$
\mathbb{P}(C = \ell) \sim \Phi\left(\frac{n}{2^\ell}\right), \quad \Phi(x) := \frac{1}{Q_\infty} \sum_{k \geq 0} (-1)^k q^{k(k-1)/2} e^{-x} q^{-k} Q_k,
$$

where $Q_k := (1 - q)(1 - q^2) \cdots (1 - q^k)$ and $q = \frac{1}{2}$ for binary case.

Count till $N$ using $\log_2 \log N + \delta$ bits, with accuracy $\sim 0.59 \cdot 2^{-\delta/2}$.
Beats information theory: 8 bits for counts $\leq 2^{16}$ w/ accuracy $\approx 15\%$. 
Recurrences: \( P_{n+1,\ell} = (1 - q^\ell)P_{n,\ell} + q^{\ell-1}P_{n,\ell-1} \).
\[ E_n(2^C) = n + 2, \quad V(2^C) = n(n + 1)/2 \] (Morris 1977).

**Symbolic methodology:**
(i) Describe events; (ii) translate to generating functions (GFs).
An alphabet \( \mathcal{A} \) with weights for Bernoulli trials. For a language describing an event \( \mathcal{E} \), the GF is

\[
E(z) \equiv \sum_n E_n z^n = \sum_n P_n(\mathcal{E})z^n
\]

<table>
<thead>
<tr>
<th>( a \in \mathcal{A} )</th>
<th>( \mapsto )</th>
<th>( \alpha z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{E} \cup F )</td>
<td>( \mapsto )</td>
<td>( E(z) + F(z) )</td>
</tr>
<tr>
<td>( \mathcal{E} \odot F )</td>
<td>( \mapsto )</td>
<td>( E(z) \times F(z) )</td>
</tr>
<tr>
<td>( \mathcal{E}^* )</td>
<td>( \mapsto )</td>
<td>( (1 - E(z))^{-1} )</td>
</tr>
</tbody>
</table>

\[
\frac{1}{1-f} = 1 + f + f^2 + \cdots \approx (f)^*
\]

\[ a_1^* \cdot b_1 \cdot a_2^* \cdot b_2 \cdot a_3^* \cdot b_3 \]
Perform probabilistic valuation $a_j \mapsto q^j$; $b_j \mapsto 1 - q^j$:

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

Do partial fraction expansion to get exact probabilities.

Do $(1 - a)^n \approx e^{-na}$ to get main approximation:

$$\mathbb{P}(C = \ell) \sim \Phi \left( \frac{n}{2\ell} \right), \quad \Phi(x) := \frac{1}{Q_\infty} \sum_{k \geq 0} (-1)^k q^{k(k-1)/2} \frac{e^{-xq^k}}{Q_k},$$

where $Q_k := (1 - q)(1 - q^2) \cdots (1 - q^k)$, and $q = \frac{1}{2}$ for binary case.

Dyadic superpositions of models: \( P_n(C = \ell) \sim \Phi(n/2^\ell) \).

\[
\text{Mean}(X) - \log_2 n \quad \longrightarrow \quad E_n(C) \sim \sum_\ell \ell \Phi\left(\frac{n}{2^\ell}\right)
\]

**Real analysis** is possible: Knuth 1965, Guibas 1977+, Fill-Mahmoud-Szpankowski-Janson, Robert-Mohamed, ... 

- **Complex asymptotic methodology:** **Mellin transform** (FlDuGo95, FlSe*)

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

Need **singularities in complex plane**.

Mellin: **Probabilistic counting, loglog counting** + Lempel-Ziv compression (Jacquet-Szpa) + dynamic hashing + tree protocols (Jacquet+) + Quadtries &c.
Mellin transform $f^*(s) = \int_0^\infty f(x) \, dx$, from real to complex.

♥ Maps asymptotics of $f$ at $0$ and $+\infty$ to singularities of $f^*$ in $\mathbb{C}$:

$$C \cdot x^\alpha \xrightarrow{\mathcal{M}} \pm \frac{C}{s + \alpha}.$$~~

Reason: Inversion theorem $\frac{1}{2i\pi} \int_{c - i\infty}^{c + i\infty} f^*(s)x^{-s} \, ds + \text{Residues}$.

♥ Factorizes harmonic sums:

$$\sum \lambda \cdot f(\mu x) \xrightarrow{\mathcal{M}} f^*(s) \cdot \sum \frac{\lambda}{\mu^s}.$$~~

For dyadic sums: $\sum f(x2^{-k}) \xrightarrow{\mathcal{M}} \frac{f^*(s)}{1 - 2^s}$

$$\alpha = 2i\kappa \pi / \log 2 \implies x^{-\alpha} = e^{-2i\kappa \pi \log_2 x}.$$
Cultural flashes

— **Complexity**: Morris (1977): Counting a large number of events in small memory. The power of probabilistic machines & approximation (Freivalds 1977).

— **Special functions**: Mellin analysis involves partition identities for Dirichlet series. Prodinger has connections with *q*-hypergeometric functions.

\[
\sum_{n \geq 0} q^{n(n+1)/2} \frac{x^n w^n}{(1 + xq) \cdots (1 + xq^{n+1})} = \sum_{n \geq 0} (-qx)^n \left[ (1 - w) \cdots (1 - q^{n-1}w) \right].
\]

— **Probability theory**: Exponentials of Poisson processes (Yor et al).

\[
\sum_i E_i q^i, \text{ where } E_i \in \text{Exp}(1).
\]

— **Communication**: The TCP protocol = Additive Increase Multiplicative Decrease (AIMD) leads to similar functions (Robert et al, 2001).

Ethernet: Get waiting time for a packet subject to \(k\) collisions (Robert).

Ethernet is unstable (Aldous 1986) but tree protocols are stable (Jacquet+).
2 CARDINALITY ESTIMATORS

Given stream (read-once sequence), estimate number of distinct elements.

— Adaptive sampling
— Probabilistic Counting
— LogLog Counting
2.1 Adaptive Sampling

- An algorithm of M. Wegman (1980+) that does *cardinality estimation* for \( s = s_1 \ldots s_\ell \) and more:

  Samples *uniformly over domains (sets)* of multisets = of independent interest for data bases.


DataBases: Given \( \{ \text{persons, towns} \} \), get geography from demography?

(© Bettina Speckmann, TU Eindhoven)
Sample values (i.e., without multiplicity)?

**Algorithm:** Adaptive Sampling (without multiplicities)

/* Get a sample of size \( \leq m \) according to distinct values. */

— **On overflow:** Increase sampling depth and decrease sampling rate = use farther bits to filter.

Sample of size \( \leq m \):

depth \( d = 0, 1, 2, \ldots \)

Analysis makes use of digital trees, generating functions and Mellin transforms.
First Counting Algorithm for cardinalities.

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

**Theorem:** $X := 2^d \xi$ estimates the cardinality of $S$ using $m$ words:

- It is unbiased: $E_n(X) = n$.
- Standard error is $\sim \frac{1}{\sqrt{(m-1) \log 2}} \approx \frac{1.20}{\sqrt{m}}$.
- Distribution is a Louchard compound [Louchard00].

With $m = 1,000W$, get 4% accuracy.

- Related to folk algorithm for leader election on channel: "Talk, flip coin if noisy; sleep if Tails; repeat!"
- Related to “tree protocols with counting” $\gg$ Ethernet. Cf (Greenberg-F-Ladner JACM 1987).
Analysis: Digital tree aka trie, paged version:

\[
\text{Trie}(\omega) \equiv \omega \text{ if } \text{card}(\omega) \leq b
\]

\[
\text{Trie}(\omega) = \begin{cases} 
    \text{Trie}(\omega \setminus 0) & \text{if } \text{card}(\omega) > b \\
    \text{Trie}(\omega \setminus 1) & \text{if } \text{card}(\omega) > b
\end{cases}
\]

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

Refs: (Knuth Vol 3), (Sedgewick, Algs), (Mahmoud), (Szpankowski**). B. Vallée’s
dynamical sources; Bentley-Sedge trees + (Clément-F-Vallée), (Devroye*), etc.
For recursively defined parameters: $\alpha[\omega] = \tau[\omega] + \beta[\omega \setminus 0] \cdot \gamma[\omega \setminus 1]$:

$$E_n(\alpha) := E_n(\tau) + \sum_{k=0}^{n} \frac{1}{2^k} \binom{n}{k} E_k(\beta) \cdot E_{n-k}(\gamma).$$

**Exponential generating functions (EGF):** $A(z) := \sum_{n} E_n(\alpha) \frac{z^n}{n!}$

$$A(z) = T(z) + B(z) \cdot C(z).$$

For (left) recursive parameter $\phi$:

$$\Phi(z) = e^{z/2} \Phi\left(\frac{z}{2}\right) + \text{Toll}(z)$$

Solve by iteration, extract coefficients; Mellin-ize!

*More in AofA talks by Szpankowski & Devroye!*
Hamlet

• **Straight Sampling** (13 elements):
  
  \[
  \text{Google} \left( \text{leaue} \mapsto \text{leave}, \text{ophe} \mapsto \emptyset \right) = 38,700,000.
  \]

  ———

• **Adaptive Sampling** (10 elements):
  
  danskers, distract, fine, fra, immediately, loses, martiall, organe, pas- 
  seth, pendant

  \[
  \text{Google} = 8, \text{all are to Shakespeare's Hamlet} \leadsto \text{mice, later!}
  \]
2.2 Probabilistic Counting

Second Counting Algorithm for cardinalities:

**Algorithm:** Probabilistic Counting

Input: a stream \( S \); Output: cardinality \(|S|\)

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

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

Return \( P \) where \( P \) is position of first 0.
Lemma: $P$ estimates $\log_2(\varphi n)$, with $\varphi \doteq 0.77351$.

— **Straight averaging** over $m$ trials, $\text{Ave} = \frac{1}{m}[P_1 + \cdots + P_m]$; return $\frac{1}{\varphi}2^{\text{Ave}}$; expect error $O(1/\sqrt{m})$.

— **Stochastic averaging** = one hash function and $O(1)$ per record:

Split (mentally) stream: e.g., $S \mapsto (S_{000}, \ldots, S_{111})$, for $m = 8$;
Work out each $P_j := P(S_j)$ separately; /* cost $O(1)$ per element */
Let $\text{Ave} := \frac{1}{m}[P_1 + \cdots + P_m]$; /* used to estimate $\frac{n}{m}$ */
Return $\frac{m}{\varphi}2^{\text{Ave}}$. 
**Theorem [FM85]:** Define *magic constant* $\varphi$ as

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

— Probabilistic Counting is *asymptotically unbiased* (up to $10^{-5}$ fl.).

— Accuracy is $\frac{0.78}{\sqrt{m}}$ for $m$ Words of size $\log_2 N$.

— $\exists$ asymptotic form of distributions w/ exponential tails.

E.g. $1,000W = 4$kbytes $\sim 2.5\%$ accuracy.
Proof: trie analysis

\[
1 \cdot (e^{x/8} - 1)(e^{x/4} - 1)(e^{x/2} - 1) \\
(1 - q)(1 - q^2)(1 - q^4) \cdots = \sum_n (-1)^\sum \text{bits}(n) q^n.
\]

**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.
Application: Data mining of the Internet graph
(Palmer, Gibbons, Faloutsos², Siganos 2001)

Internet graph: 285k nodes, 430k edges.

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

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

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

\[\text{b) Histogram of diameters}\]

+ Sliding window usage (Motwani et al).
2.3 LogLog Counting

Third Counting Algorithm for cardinalities: (Durand-F, 2003/DFFGM, 2006)

Claim: WAS/IS the best algorithm on the market!

• Hash and get $\rho(h(x)) := \text{position of rightmost 1-bit} \approx \text{a geometric RV.}$
• To set $S$ associate $R(S) := \max_{\nu \in S} \rho(h(\nu)).$

- Max of geometric RVs are well-known (Prodinger*).

$R(S)$ estimates $\sim \log(\hat{\phi} \ \text{card}(S)),$ with $\hat{\phi} := e^{-\gamma \sqrt{2}}.$
Algorithm LogLog Counting:

- Use rightmost 1-bit as “observable”.
- Do stochastic averaging with \( m = 2^\ell \). E.g., \( S \equiv \langle S_{00}, S_{01}, S_{10}, S_{11} \rangle \).
- Return: \( \frac{m}{\hat{\phi}} 2^{\text{Average}} \), where \( \hat{\phi} = e^{-\gamma} \sqrt{2} \).

+ Switch to Hit Counting for small cardinalities.
++ Optimize by pruning discrepant values \( \sim \) superLogLog or better by harmonic means \( \sim \) FFGM’s HyperLogLog (\( \ll \) Chassaing-Gerin, 2006)
**Theorem.** LogLog is asymptotically unbiased.

— It needs $m$ “bytes”, each of length $\log_2 \log N_{\text{max}}$.

— Standard error (accuracy) is: $\frac{1.30}{\sqrt{m}}$, where $1.30 \approx \sqrt{\frac{1}{12} \log^2 2 + \frac{1}{6} \pi^2}$.

— Distribution is approximately Gaussian.

Whole of *Shakespeare*:

$m = 256$ small “bytes” of 4 bits each $\equiv 128\text{bytes}$

```
ghffghfghghgghhheehffhhgghgghffffhhiigfhhffgfihihhhh
igigighgfihffghigihghigfhheegeghghghhhhfgfhhidiihihihihehhhfgg
hfgighfgfghdieghhhhggghffghhiheffgghgihfgggffihgihfgghggiiff
fjfgjhhjiifhjgehghffhfhjhigghghgihhiihiihgiighgfhlgjfgjjmjf
```

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

Error is $+9.4\%$ for 128 bytes(!!)
Proof: Trie-like analyses: need \( \text{coeff}[z^n] \left( e^z \sum_k 2^{k/m} \left[ e^{-z/2^k} - e^{-z/2^k} \right] \right)^m \).

**Analytic depoissonization [JaSz95+]**

- Recover asymptotics of \( f_n \) from \( \phi(z) := \sum_n f_n e^{-z} \frac{z^n}{n!} \equiv \text{Poisson GF?} \)
- Intuition: with luck \( f_n \sim \phi(n) \)

Here: “Luck” means good lifting of \( \phi(z) \) to \( \mathbb{C} \equiv \text{Poisson flow of complex rate!} \)
E.g.: \( \exists \text{cone. Inside: } \phi(z) \sim z^\alpha \). Outside: \( \phi(z) \) is exponentially smaller.

\[
f_n = \frac{n!}{2i\pi} \oint e^z \phi(z) \frac{dz}{z^{n+1}} \approx \phi(n)
\]
**Features:** Errors $\approx$ *Gaussian*, seldom more than $3 \times$ standard error. Algorithm *scales down* and up (for small/large cardinalities).

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

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

<table>
<thead>
<tr>
<th>m</th>
<th>$2^6$ (50by)</th>
<th>$2^{10}$ (0.8kby)</th>
<th>$2^{14}$ (12kb)</th>
<th>$2^{18}$ (200kb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obs:</td>
<td>8.9%</td>
<td>2.6%</td>
<td>1.2%</td>
<td>0.32%</td>
</tr>
<tr>
<td>$\sigma$:</td>
<td>11%</td>
<td>2.8%</td>
<td>0.7%</td>
<td>0.36%</td>
</tr>
</tbody>
</table>
Summary of $F_0$ algorithms

$N = 10^8$ & 2% error

— Hit Counting: $\approx \frac{1}{10} N_{\text{max}} = 1 \text{ Mbyte}$ + used for corrections
— Adaptive Sampling ($\epsilon = \frac{1.20}{\sqrt{m}}$): 16 kbytes + domain sampling, mice
— Probabilistic Counting ($\epsilon = \frac{0.78}{\sqrt{m}}$) = 8 kbytes + sliding window
— Multiresolution bitmap (analysis??) = 5 kbytes?
— MinCount ©Giroire = 4 kbytes + sliding window
— Loglog Counting ($\epsilon = \frac{1.30}{\sqrt{m}}$) = 2 kbytes + elephants

Document similarity

An application of cardinality counts. For multisets $A$ and $B$, define
$$\text{sim}(A, B) := \frac{|A \cap B|}{|A \cup B|} \quad \text{(Broder)}.$$ 

Here: $|A| := \text{card}(A)$. Let $\text{Reg}(A)$ be signature of $A$, i.e., (LogLog) register dump of $A$, so that $|A| = \text{estim}(\text{Reg}(A))$.

$$\begin{align*}
|A| &= \text{estim}(\text{Reg}(A)); \\
|B| &= \text{estim}(\text{Reg}(B)) \\
|A \cup B| &= \text{estim} \left( \text{Reg}(A) \bigoplus_{\max} \text{Reg}(B) \right); \\
|A \cap B| &= |A \cup B| - |A| - |B|.
\end{align*}$$

- For $r$ files, pairwise comparisons have cost $O(\sum |F_j|) + O(r^2)$, as opposed to $O(\sum |F_j|)^2$:
  $$\implies \text{For } 10^5 \text{ files of size } 10^5, \text{ work in minutes instead of days!}$$
A blind test, by Pranav Kashyap

39 files of 20k words each, encrypted word-by-word.

- How many languages? Which are which?
Blind classification ($\theta = 0.25$)

Actual ($\theta = 0.20$)
3 MICE

Simply use **Adaptive Sampling** and keep running counts!

— Hamlet: catch the frequency profile of *mice*:
danskers\(^1\), distract\(^1\), fine\(^9\), fra\(^1\), immediately\(^1\), loses\(^1\), martia\(^1\), organe\(^1\), passeth\(^1\), pendant\(^1\).

— With *cache size* = 100, get a sample of 79 elements. 
\[1^{50}, 2^{14}, 3^{4}, 4^{2}, 5^{1}, 6^{1}, 9^{1}, 13^{1}, 15^{1}, 28^{1}, 43^{2}, 128^{1}\].

<table>
<thead>
<tr>
<th></th>
<th>1-Mice</th>
<th>2-Mice</th>
<th>3-Mice</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Estimated</em></td>
<td>63%</td>
<td>17%</td>
<td>5%</td>
</tr>
<tr>
<td><em>Actual</em></td>
<td>60%</td>
<td>14%</td>
<td>6%</td>
</tr>
</tbody>
</table>

The 10 most frequent words in Hamlet are: the, and, to, of, i, you, a, my, it, in. They account for > 20% of all text. With 20 words, capture 30%; with 50, get 44%. **70 words capture 50% of all occurrences!**
Definition: A k-iceberg is present in proportion > 1/k.

Algorithm Majority/Icebergs: /* For k = 2, uses 1 registers */
— Trigger a gang war: equip each individual with a gun.
— Two guys from different gangs shoot and kill one another. Majority gang is only surviving one.
— Adapt to k ≥ 2 with k – 1 registers. Supplement with straight sampling, say, with m = 7k.

(Karp, Shenker, Papadimitriou + Bar-Yossef et al. 2001-2002)
Here: largest elephants (= to given destination) as function of time.

RAP @ Inria: Bloom filters (Azzana, Chabchoub, Ph. Robert)

E.g. Think of 1Billion records; 10Million are 1-mice; Others are 100–10,000 elephants. No implied locality. What to do with memory = 1kWords?
Counting elephants via cardinality algorithms

= A cute idea of O. Gandouet and A. Jean-Marie @ Montpellier.

**Algorithm** GJM’s ElephantCount

— Let $S$ be stream composed (say) of 1-mice and $\geq 100$ elephants.
— Estimate $N := |S|$ and $N_0 := |S_0|$, with $S_0$ a prob. $p$-sample of $S$.
— Solve system

$$
\begin{align*}
N &= N_m + N_e \\
N_0 &= p \cdot N_m + 0.999^+ \cdot N_e
\end{align*}
$$

, with $p = \frac{1}{10}$ (say).
Counting triangles in graphs

Suggested by (Bar-Yossef, Kumar, Sivakumar, 2002) who propose to use $F_2$ (!)

Consider graph $\Gamma$ of max-degree $D$ given by adjacency list.

- Define a vee ($\vee$) as any triple $\{u, v, w\}$ such that
- Make a stream of all vee’s with cost $O(nD^2)$.
- Isolated vee’s are 1-mice; triangles are 3-elephants.
- Use cardinality on top of $p$-sampling: e.g., $p = \frac{1}{2}$ gives

$$N = N_m + N_e, \quad N_0 = \frac{1}{2} N_m + \frac{7}{8} N_e.$$
6 FREQUENCY MOMENTS: $F_2$

Recall: Alon, Matias, Szegedy (STOC 1996)*** $F_2 := \sum_v (f_v)^2$, where $f_v$ is frequency of value $v$. An elegant idea: $\text{flip}(x) \equiv \epsilon(x) = \pm 1$ based on $\text{hash}(x) = \text{"reproducible randomness"}$.

**Alg:** $F_2$;
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.

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

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

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

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

$c_1 X_1 + c_2 X_2 \iid \mu X$, 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(?)
Conclusions

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

— Sample distinct values;
— Estimate $F_0, F_1, F_2, F_p$ ($0 < p \leq 2$) even for huge data sets;
— Estimate icebergs, # of mice and elephants.

♥ Need virtually no assumption on nature of data.

♥♥♥♥♥♥♥

The algorithms are based on randomization $\rightarrow$ 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 depoissonization, Mellin transforms, stable laws and Mittag-Leffler functions, etc.”
That’s All, Folks!