Data Streams as Random Permutations: the Distinct Element Problem

Dedicated to the memory of Philippe Flajolet (1948-2011)

Conrado Martínez,
Univ. Politècnica de Catalunya, Barcelona, Spain

AofA, Montréal, June 2012

Joint work with:

A. Helmi  J. Lumbroso  A. Viola
A data stream is a (very long) sequence

\[ S = s_1, s_2, s_3, \ldots, s_N \]

of items \( s_i \) drawn from some (large) domain \( \mathcal{U} \), \( s_i \in \mathcal{U} \)

The goal: to compute \( y = y(S) \), but ...
A data stream is a (very long) sequence

\[ S = s_1, s_2, s_3, \ldots, s_N \]

of items \( s_i \) drawn from some (large) domain \( \mathcal{U} \), \( s_i \in \mathcal{U} \)

The goal: to compute \( y = y(S) \), but …
Introduction

...there are limitations to our computational power:

- a single pass over the sequence
- very short time for computation on each item
- very small auxiliary memory: $M \ll N$; ideally $M = \Theta(1)$ or $M = o(\log N)$
- no statistical hypothesis on the data
... there are limitations to our computational power:

- a single pass over the sequence
- very short time for computation on each item
- very small auxiliary memory: $M \ll N$; ideally $M = \Theta(1)$ or $M = \mathcal{O} \left( \log N \right)$
- no statistical hypothesis on the data
there are limitations to our computational power:

- a single pass over the sequence
- very short time for computation on each item
- very small auxiliary memory: $M \ll N$; ideally $M = \Theta(1)$ or $M = \Theta(\log N)$
- no statistical hypothesis on the data
there are limitations to our computational power:
- a single pass over the sequence
- very short time for computation on each item
- very small auxiliary memory: $M \ll N$; ideally $M = \Theta(1)$ or $M = \mathcal{O}(\log N)$
- no statistical hypothesis on the data
There are lots of applications for this data stream model:

- Network traffic analysis $\Rightarrow$ DoS/DDoS attacks, worms, ...
- Database query optimization
- Information retrieval $\Rightarrow$ similarity index
- Data mining
- And many more . . .
We will often see $S$ as a multiset

$$\{w_1 \circ f_1, \ldots, w_n \circ f_n\},$$

with

$$f_i = \text{frequency of the } i\text{th distinct element } w_i$$
Introduction

Some typical problems:

- The cardinality of $S$: $\text{card}(S) = n \leq N \iff \text{This paper}$
- Frequency moments $F_p = \sum_{1 \leq i \leq n} f_i^p$
  (N.B. $n = F_0$, $N = F_1$)
- The elements $\nu_i$ such that $f_i \geq k$ ($k$-elephants)
- The elements $\nu_i$ such that $f_i < k$ ($k$-mice)
- The elements $\nu_i$ such that $f_i \geq cN$, $0 < c < 1$ ($c$-icebergs)
- The $k$ most frequent elements
- ...
Small auxiliary memory ⇒

Exact solution too costly (or impossible) ⇒

Randomized algorithms ⇒

Estimation \( \hat{y} \) of the quantity \( y \)

- The estimator \( \hat{y} \) must be unbiased
  \[
  E[\hat{y}] = y
  \]

- The estimator must be accurate (small standard error)
  \[
  SE[\hat{y}] := \frac{\sqrt{\text{Var} [\hat{y}]}}{E[\hat{y}]} < \epsilon,
  \]
  e.g., \( \epsilon = 0.01 \) (1%)
Late in the 70s, G. Nigel N. Martin invents probabilistic counting, for database query optimization. He detects systematic bias in his estimator, he tweaks the algorithm to correct the bias.
When Flajolet learns about the algorithm, he contacts Martin and they team up to carry out a very detailed analysis giving the correcting factor and upper bounds for the standard error.

Their pioneering work (Flajolet & Martin, JCSS, 1985) introduces many of the ideas behind the most practical and successful cardinality estimators.
Estimating the cardinality

The first ingredient:

- Map each item $s_i$ to a value in $(0, 1)$ using a hash function $h: \mathcal{U} \rightarrow (0, 1)$ ⇒ reproducible randomness
- The multiset $S$ is mapped to a multiset

$$S' = h(S) = \{x_1 \circ f_1, \ldots, x_n \circ f_n\},$$

with $x_i = \text{hash}(w_i)$, $f_i = \# \text{ of } x_i$’s

- The set of distinct elements $X = \{x_1, \ldots, x_n\}$ is a set of $n$ independent and uniformly distributed real numbers in $(0, 1)$
Estimating the cardinality

The first ingredient:

- Map each item $s_i$ to a value in $(0, 1)$ using a hash function*
  \[ h : \mathcal{U} \rightarrow (0, 1) \Rightarrow \text{reproducible randomness} \]
- The multiset $S$ is mapped to a multiset
  \[ S' = h(S) = \{ x_1 \circ f_1, \ldots, x_n \circ f_n \}, \]
  with $x_i = \text{hash}(w_i), f_i = \# \text{ of } x_i$’s
- The set of distinct elements $X = \{ x_1, \ldots, x_n \}$ is a set of $n$ independent and uniformly distributed real numbers in $(0, 1)$

*We disregard here collisions: if the hash values have enough bits the probability of collision can be neglected
Probabilistic Counting

The second ingredient:

- Define some easily computable observable $R$ which is insensitive to repetitions, that is, it only depends on the underlying set of distinct elements:

$$R = R(S) = R(X)$$

- Perform the probabilistic analysis of $R$ for a set $X$ of $n$ random real numbers. If

$$E_n[R] = \varphi(n)$$

then it is reasonable to assume that the expected value of $\varphi^{-1}(R)$ will be close to $n$; we will need some correcting factor $\kappa$ to get an (asymptotically) unbiased estimator

$$E_n\left[\kappa\varphi^{-1}(R)\right] = n + \text{l.o.t.}$$
Probabilistic Counting

- For instance, in Flajolet & Martin’s Probabilistic Counting the observable \( R \) is the length of the longest prefix \( 0.0^{R-1}1 \) such that all prefixes \( 0.0^k1 \) appear among the hashed values, for \( 0 \leq k \leq R - 1 \)
- \( R \) is easy to compute and it does not depend on repetitions

\[
E_n[R] \approx \log_2 n
\]

and

\[
E_n[\kappa 2^R] = n + o(n)
\]

for

\[
\kappa^{-1} = \frac{e^{\gamma} \sqrt{2}}{3} \prod_{k \geq 1} \left( \frac{(4k + 1)(2k + 1)}{2k(4k + 3)} \right)^{(-1)^{\nu(k)}} \approx 0.77351 \ldots
\]
Other estimators

- **LogLog** (Durand, Flajolet, 2003) and **HyperLogLog** (Flajolet, Fusy, Gandouet, Meunier, 2007) use bit patterns in the hash values to estimate, like in Probabilistic Counting.

- **Order statistics** (e.g., the $k$th smallest in the set of distinct hash values) have also been used to estimate cardinality: Bar-Yossef, Kumar & Sivakumar (2002); Bar-Yossef, Jayram, Kumar, Sivakumar & Trevisan (2002); Giroire (2005, 2009); Chassaing & Gérin (2006); Lumbroso (2010).
**Recordinality**

- **Recordinality** counts the number of records (more generally, $k$-records) in the sequence.
- It depends in the underlying permutation of the first occurrences of distinct values, very different from the other estimators.
- If we assume that the first occurrences of distinct values form a random permutation then **no need for hash values**!
Recordinality

- $\sigma(i)$ is a record of the permutation $\sigma$ if $\sigma(i) > \sigma(j)$ for all $j < i$

- This notion is generalized to $k$-records: $\sigma(i)$ is a $k$-record if there are at most $k - 1$ elements $\sigma(j)$ larger than $\sigma(i)$ for $j < i$; in other words, $\sigma(i)$ is among the $k$ largest elements in $\sigma(1), \ldots, \sigma(i)$
procedure RECORDINALITY(\(S\))

fill \(T\) with the first \(k\) distinct elements (hash values) of the stream \(S\)

\(R \leftarrow k\)

for all \(y \in S\) do

\(x \leftarrow h(y)\)

if \(x > \min(T) \land x \not\in T\) then

\(R \leftarrow R + 1; T \leftarrow T \cup \{x\} \setminus \min(T)\)

end if

end for

end procedure

Memory: \(k\) hash values (\(k \log n\) bits) + 1 counter (\(\log \log n\) bits)
**Theorem (Helmi, Martínez and Panholzer)**

Let $r_k$ denote the number of $k$-records in a permutation of size $n$. The exact distribution of $r_k$ is

$$
Prob_n \{r_k = j\} = \begin{cases} 
[n = j] & \text{if } k > n, \\
\frac{k^{j-k} k!}{n!} \left[\frac{n-k+1}{j-k+1}\right] & \text{if } k \leq j \leq n
\end{cases}
$$

$$\left[\begin{array}{c}
n \\
j
\end{array}\right] = \text{signless Stirling numbers of the first kind}; \left[\begin{array}{c}P \\
\end{array}\right] = 1 \text{ if } P \text{ true, } = 0 \text{ otherwise}$$
The expected value of $r_k$ is $k \log(n/k) + \text{l.o.t.}$; it is reasonable then to assume that for

$$Z := k \exp(\phi \cdot r_k)$$

we should have $E_n [Z] \sim n$ for some suitable correcting factor $\phi$

We can use the formula for $\text{Prob}_n \{r_k = j\}$ to explicitly compute $E_n [Z]$ and to determine $\phi$, and then compute the standard error
Theorem

The RECORDINALITY estimator

\[ Z := k \left( 1 + \frac{1}{k} \right)^{r_k - k + 1} - 1 \]

is an unbiased estimator of \( n \): \( E_n [Z] = n \).
The accuracy of **RECORDINALITY**, expressed in terms of standard error, asymptotically satisfies

\[ SE_n [Z] \sim \sqrt{\left( \frac{n}{ke} \right)^{\frac{1}{k}}} - 1 \]
For practical values of $n$, even for small $k$, the estimates may be significantly concentrated.

For instance, for $k = 10$, the estimates are within $\sigma$, $2\sigma$, $3\sigma$ of the exact count in respectively 91%, 96% and 99% of all cases.

$k = 64$

$k = 256$

500 estimates of cardinality in Shakespeare’s *A Midsummer Night’s Dream*; top and bottom lines (5%), centermost lines (70%); gray area (1 standard deviation)
- **RECORDINALITY** does not depend on the hash values, only the relative ordering ⇒ we can avoid using the hash function, provided the distinct elements appear (for the first time) in random order.

- We can combine **RECORDINALITY** with any of the other $k$th order statistic estimators since they are independent; we can get both estimators with a single pass of the “scanning” algorithm.
Other issues

- The table of $k$th largest hash values gives us a random sample of $k$ distinct elements out of the $n \Rightarrow \text{distinct sampling}$ for free.

- If we keep all distinct $k$-records, not just the $k$ largest distinct values, we have a random sample of expected size $k \log(n/k) \Rightarrow \text{variable-size sampling}$!
Concluding remarks

- First (?) application of combinatorics of random permutations to data stream algorithms
- Simple and elegant algorithms
- Nice combinatorics and mathematical analysis
- Many extensions to explore: sampling, sliding windows, similarity index, ....
Thanks a lot for your attention!