Sketching Probabilistic Data Streams

Graham Cormode
AT&T Labs - Research
graham@research.att.com

Minos Garofalakis
Yahoo! Research
minos@acm.org
Challenge of Uncertain Data

- Many applications generate data which is uncertain:
  - Quality of Record Linkages
  - Confidences of extracted rules
  - Noisy Sensor/RFID readings
- Leads us to study *probabilistic data management*
- *Tuple level uncertainty*: each tuple is uncertain, independent
- Leads to exponentially many *possible worlds*
Simple Model

- We adopt simplest model (Dalvi and Suciu 2004):
  - A set of probabilistic tuples \( \langle t, p \rangle \)
  - A pair of a value, \( t \in [1 \ldots M] \) and a probability \( p \)
  - With probability \( p \), \( t \) is in relation, \( (1-p) \) it is not
  - More generally, can have a (compact) PDF

- Example: \( S = \langle \langle x, \frac{1}{2} \rangle, \langle y, \frac{1}{3} \rangle, \langle y, \frac{1}{4} \rangle \rangle \)
  - Encodes 6 “possible worlds” ground relations: \( \text{Grnd}(S) = \{ \phi, (x), (y), (x, y), (y, y), (x, y, y) \} \)
  - Can compute probabilities of each possible relation:

<table>
<thead>
<tr>
<th>G</th>
<th>( \phi )</th>
<th>( x )</th>
<th>( y )</th>
<th>( x,y )</th>
<th>( y,y )</th>
<th>( x,y,y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Pr}[G] )</td>
<td>( \frac{1}{4} )</td>
<td>( \frac{1}{4} )</td>
<td>( \frac{5}{24} )</td>
<td>( \frac{5}{24} )</td>
<td>( \frac{1}{24} )</td>
<td>( \frac{1}{24} )</td>
</tr>
</tbody>
</table>
In general, too expensive to track all possible worlds

Probabilistic streams: too expensive to track all tuples!
  - E.g. stream of sensor readings

Want to compute aggregate functions over prob. streams
  - Given function $F$, find expected value:
    \[
    E(F(S)) = \sum_{G \in \text{Grnd}(S)} \text{Pr}[G] \, F(G)
    \]
  - Also compute variance to quantify reliability:
    \[
    \text{Var}(F(S)) = E(F^2(S)) - E^2(F(S))
    \]

Focus on computing Frequency moments ($F_0$, $F_1$, $F_2$), much studied in deterministic streams

Measure space and time cost (one pass over stream)
Sampling Approach

Efficient streaming algorithms are known for many deterministic streaming computations

Natural idea: sample several possible ground streams, compute $F$ on each, and compute $E$ and $\text{Var}$ of samples.

– Can work OK for $E$: sampling $O(\varepsilon^{-2} \text{Var}[F(S)]/E^2[F(S)])$ gives relative error $\varepsilon$.
– Depends on the stream and aggregate properties, but for many cases, the ratio $\text{Var}/E^2$ is small.
– Bounds for estimating $\text{Var}$ are much worse, need many more samples
Warm up case: $F_1$

- Some functions are easy to compute exactly, in streaming model with small cost.
- $F_1$ is just count – $E(F_1(S))$ is expected length of stream
- Easy to see $E(F_1(S)) = \sum_i p_i$ (sum of Bernoulli variables)
- By summation of variances, $\text{Var}(F_1(S)) = \sum_i p_i(1-p_i)$

- Can use these observations to estimate quantiles and heavy hitters with additive error $\varepsilon$ in space $O(1/\varepsilon)$
**F₀: Count Distinct**

- \( \mathbb{E}[F_0(S)] \) is the expected number of distinct tuples seen
  - Easy to track in (high) space \( O(M) \), by keeping information for each possible tuple value \( t \).

- \( M \) is often very large, want solution with cost \( O(\log M) \)
  - Make use of the Flajolet-Martin (FM) sketch, which solves \( F_0 \) for deterministic streams
FM Sketch Summary

- Estimates number of distinct inputs (count distinct)
- Uses hash function mapping input items to \( i \) with prob \( 2^{-i} \)
  - i.e. \( \text{Pr}[h(x) = 1] = \frac{1}{2}, \text{Pr}[h(x) = 2] = \frac{1}{4}, \text{Pr}[h(x)=3] = \frac{1}{8} \ldots \)
  - Easy to construct \( h() \) from a uniform hash function by counting trailing zeros
- FM Sketch = bitmap array of \( L = \log M \) bits
  - Initialize bitmap to all 0s
  - For each incoming value \( x \), set \( \text{FM}[h(x)] = 1 \)

\[
\begin{align*}
x = 5 & \quad \Rightarrow \quad h(x) = 3 \\
\text{FM BITMAP} & \quad \begin{array}{ccccccc}
0 & 0 & 0 & 1 & 0 & 0 & 0
\end{array}
\end{align*}
\]
In FM sketch, 1 indicates that some item hashed there
Interpret this as a probability: when $t_i$ arrives, update
$$p_{FM}[h(t_i)] \leftarrow p_i + (1-p_i)p_{FM}[h(t_i)]$$

Build estimator $D$ for $E[F_0(S)]$ as
$$D = \sum_j 2^j p_{FM}[j] \prod_{k>j} (1-p_{FM}[k])$$
  - uses (expected) location of most significant 1 bit in array

Can show that $D$ is a constant factor approx of $E[F_0(S)]$
with constant probability
Improved Estimator

- Can build an \((\varepsilon, \delta)\) estimator for \(E[F_0(S)]\): finds a value \(d\) such that \(d = (1 \pm \varepsilon) E[F_0(S)]\) with probability at least \(1-\delta\)
  - Using same pFM sketch as before

- Use constant factor approx to find a sampling level
  \(k^* \approx \log_2 E[F_0(S)] + O(1)\)
  - Probe multiple repetitions of sketch at level \(k^*\) to build a better estimator (details in paper)

- Can \((\varepsilon,\delta)\) approximate \(E[F_0(S)]\) using \(O(\varepsilon^{-2} \log \delta^{-1})\) pFMs
  - Similar to cost for deterministic streams
Estimating Var(F₀)

- Also want to estimate Var[F₀(S)], the variance of F₀
  - Reduce to computing E[F₀(S)] over modified streams

- Given S = (⟨tᵢ, pᵢ⟩), set S₂ = (⟨tᵢ, 2pᵢ − pᵢ²⟩)
  - Can prove that Var[F₀(S)] = E[F₀(S₂)] − E[F₀(S)]
  - Since E[F₀(S₂)] ≤ 2E[F₀(S)], error is at most 3εE[F₀(S)]

- Can estimate Var[F₀(S)] with additive error εE[F₀(S)]
  w/prob at least 1 − δ using O(ε⁻² log δ⁻¹) pFM sketches
**F₂: Self-join size**

- Let \( f_t \) be the frequency of item \( t \); \( F_2 = \sum_t f_t^2 \), self-join size.
  - On prob. streams, \( E[F_2(S)] \) is expected self-join size

- Let \( X_t \) be random variable for occurrences of \( t \).
  - \( E[X_t] = \sum_{\langle ti = t, p_i \rangle \in S} p_i \) and \( \text{Var}[X_t] = \sum_{\langle ti = t, p_i \rangle \in S} p_i (1 - p_i) \)

- Since \( E[X_t^2] = \text{Var}[X_t] + E^2[X_t] \), we have:
  - \( E[F_2(S)] = \sum_t (\sum_{\langle ti = t, p_i \rangle \in S} p_i (1 - p_i) + (\sum_{\langle ti = t, p_i \rangle \in S} p_i)^2) \)
  - First term can be computed exactly
  - Second term is a \( L^2_2 \) norm of a deterministic stream of \( p_i \)'s

- Use AMS sketch on \( p_i \)'s to \((\epsilon, \delta)\) approximate \( E[F_2(S)] \) in space \( O(\epsilon^{-2} \log \delta^{-1}) \)
Var\[F_2(S)\], variance of self-join size

- We used the fact that \( \text{Var}[X] = \text{E}[X^2] - \text{E}^2[X] \) and that \( \text{Var}[X + Y] = \text{Var}[X] + \text{Var}[Y] \) to find \( \text{E}[F_2(S)] \).

- Can use similar *cumulants* to find higher moments:
  \[
  \kappa_3[X] = \text{E}[(X - \text{E}(X))^3] \quad \kappa_4[X] = \text{E}[(X-\text{E}(X))^4] - 3\text{Var}[X]^2 \]
  and \( \kappa_j[X + Y] = \kappa_j[X] + \kappa_j[Y] \) for all \( j \).

- Can write \( \text{Var}[X^2] \) in terms of cumulants:
  \[
  \text{Var}[X^2] = \kappa_4[X] + 4\kappa_3[X]\kappa_1[X] + 2\kappa_2^2[X] + 4\kappa_2[X]\kappa_1^2[X] \]

- For Bernoulli random variable \( B \) with parameter \( p \):
  \[
  \kappa_1[B] = p, \quad \kappa_2[B] = p - p^2, \quad \kappa_3[B] = (1-2p)(p-p^2), \quad \kappa_4[B] = (1-6p+6p^2)(p-p^2) \]
**Var[F_2(S)] results**

- Consequently, can rewrite \( \text{Var}[F_2(S)] \) in terms of deterministic stream functions of the \( p_i \)'s

- Estimate in small space by using AMS sketches to estimate appropriate vector dot-products

- Can find an estimate of \( \text{Var}[F_2(S)] \) with error at most \( \varepsilon \) \( E[F_2(S)]^{3/2} \) with prob. at least \( 1-\delta \) in space \( O(\varepsilon^{-2} \log \delta^{-1}) \)

- Similar cumulant-based techniques allow estimation of join size, and higher moments
Experimental Study

- Implemented our algorithms for $F_0$ and $F_2$, both $E$ and $\text{Var}$
- Used real data from MYSTIQ project based on linkages between Amazon and IMDB data
- Synthetic data with zipfian distribution on tuples, uniform on probabilities
F₀ Results

- Sampling possible worlds for non-pathological streams does well for $E[F₀(S)]$, is terrible for $V[F₀(S)]$ (off chart)
- pFM sketches are much faster (by a factor of about 30)
F₂ Results

- Sampling slightly better on synthetic streams for expectation, still way off for variance
- Both methods fast: about 1 second to process $10^6$ tuples

80KB space
Closing Remarks

- Fundamental aggregates such as Frequency Moments can be approximated accurately on probabilistic streams
- Requires careful analysis and proof to give guarantees
- Need space and time similar to deterministic streams
- Results scale pretty well experimentally
  - e.g. 10% relative error in 80KB space
- Many other problems to study in this domain