Computational Frameworks

Streaming

Dealing with volume and velocity

Monitoring huge and rapidly changing streams of data:

- Scenarios: (sensor) network traffic, transaction logs, online trading, auctions, analyzing physical (e.g., meteorological, astronomical) events.
- Data analysis must happen on the fly and the input data are to be processed as a continuous stream (no random access to data)

Example application:

- Stream: packets routed through a router.
- Task: Gather traffic statistics (e.g., average number of connections/second to same IP address)
- Analysis must occur on the fly using limited memory (cannot store the data stream for offline processing)

The streaming model

- Sequential machine with "small" working memory. Input provided as a continuous (one-way) stream.
- **Objective of Algorithm Design:** Solve data analysis problems by inspecting the data stream in a single (or very few) passes, using working memory substantially smaller than input size (e.g., $O(1)$ or $O(\text{poly}(\log n))$ for a stream of size n).
- The use of substantially sublinear working space calls for the design of summary data structures (a.k.a. sketches)

Streaming (Recap)

The Model

- Input stream $\Sigma = x_1, x_2, \ldots x_n$ accessed/processed sequentially
- Metric 1: Size s of the working memory (aim: $s \ll n$)
- Metric 2: Number p of sequential passes over Σ (aim: $p = 1$)
- Metric 3: Processing time per item T (aim: $T = O(1)$)

Algorithm Design Techniques

- Approximate solutions (exact ones may require linear space)
- Mantain lossy summary of Σ via a synopsis data structure (e.g., random sample, hash-based sketch)

Typical data analysis tasks

- Number of distinct data items in the stream
- Frequent items
- Useful statistics: frequency moments the stream, quantiles, histograms, etc
- Optimization and graph problems: clustering, triangle counting

Many problems require extensive space to obtain exact solution: need to resort to *space/accuracy-tradeoffs*.

- Analysis uses precision parameters: e.g., $\epsilon, \delta \in (0, 1)$.
- Prove that the computed solution is within an error ϵ off the true answer with probability at least $1 - \delta$
- Working space and running time is a (non decreasing) function of $1/\epsilon$ and $1/\delta$.

Sampling/Polling

- Mantaining a random, uniform m-sample S of the data seen so far is not an immediate task: which stream items do we retain? Assume $\Sigma = x_1, x_2, \ldots x_n$ ($n = |\Sigma|$ unknown).
- Reservoir Sampling
  - Add $x_1, x_2, \ldots, x_m$ to S (w.l.o.g., $n \gg m$)
  - For $t > m$, with prob. m/t, evict random $x \in S$ and add x_t.

Theorem

Let $\Sigma = x_1, x_2 \ldots$, and let $t \geq m$. At any time t, S is a uniform m-sample of $x_1, x_2, \ldots, x_t$.

Remark: We do not need to know n. Whenever the stream ends, we get an m-sample of Σ.

Sampling/Polling (cont'd)

Proof

We show that for $1 \leq i \leq t$, $\Pr(x_i \in S) = m/t$ by induction on $t \geq m$:

- Base case $t = m$ is trivial
- For $t > m$, consider any item x_i, $i < t$ when x_t is examined:
  - $\Pr(x_i \in S | x_t \text{ not added}) = m/(t-1)$ (by inductive hp)
  - $\Pr(x_i \in S | x_t \text{ added})) = (m/(t-1))(1 - 1/m)$ (inductive hp and random eviction)
  - Using total probabilities and Bayes' rule, after step t:

$$\Pr(x_i \in S) = \left(1 - \frac{m}{t}\right) \frac{m}{t-1} + \frac{m}{t} \left[\frac{m}{t-1} \left(1 - \frac{1}{m}\right)\right] = \frac{m}{t}$$

- The proof follows, since the algoritm also ensures that

$$\Pr(x_t \in S) = \frac{m}{t}$$

Sampling-based applications: Frequent Items

- Problem: Given a stream Σ of n items and a frequency threshold $\varphi \in (0, 1)$, return all and only items in Σ that appear at least $\varphi \cdot n$ times.
- Any straightforward (sequential) strategy requires space at least linear inthe number of distinct elements in Σ (could be arbitrarily close to n)
- We can save space if we give up exactness!
- ϵ-Approximate Frequent Items (ϵ-AFI): Besides Σ and φ, let $0 < \epsilon < \varphi$. We must return:
  - All items of frequency at least $\varphi \cdot n$
  - No item of frequency smaller than $(\varphi - \epsilon) \cdot n$
- That is, we seek algorithms with no false negatives but tolerate some high-frequency false positives

Randomized Algorithm for ϵ-AFI

- Randomized algorithm: Given a precision parameter $\delta \in (0, 1)$, returns an ϵ-AFI set with probability $1 - \delta$
- Sticky Sampling: compute empirical frequencies based on a sample of the data stream
- The sampling rate depends on $n = |\Sigma|, \varphi, \epsilon$, and δ
- Assume that n is known (we will discuss how to deal with unkown n later) and let $t \equiv t(\varphi, \epsilon, \delta)$ be a value (to be fixed by the analysis).
- Sticky Sampling maintains a set of pairs $S = \{(x, f_e(x))\}$ where $x \in \Sigma$ and $f_e(x) \leq f(x)$ is an (under)estimate of x's true frequency.

Sticky Sampling (n known)

1. $S = \emptyset$
2. Examine the next element x of the Data Stream:
   - **2a** **if** $(x, f_e(x)) \in S$ **then** $f_e(x) = f_e(x) + 1$
   - **2b** **if** $(x, f_e(x)) \notin S$ **then** add $\{(x, f_e(x) = 1)\}$ to S with probability t/n (*start tracking* x *if sampled*, t to be determined by the analysis!)
3. Return all pairs in S with $f_e(x) \geq (\varphi - \epsilon)n$

Remark

Since $f_e(x) \leq f(x)$, Sticky Sampling returns no low-frequency false positives (i.e., items x with $f(x) < (\varphi - \epsilon)n$)

Sticky Sampling (n known) (cont'd)

We now prove that with probability $1 - \delta$, Sticky Sampling returns all true positives.

- Let $\{y_i : f(y_i) \geq \varphi n, 1 \leq i \leq k\}$. Clearly, $k \leq 1/\varphi$.
- Consider complementary event. By the union bound, $\Pr(\exists \text{false negative}) \leq \sum_{i=1}^{k} \Pr(f_e(y_i) < (\varphi - \epsilon)n)$
- If $f_e(y_i) < (\varphi - \epsilon)n$, then the first ϵn occurrences of y_i were not sampled! This happens with prob. $(1 - t/n)^{\epsilon n} < e^{-t\epsilon}$
- $\Pr(\exists \text{false negative}) \leq ke^{-t\epsilon} \leq (1/\varphi)e^{-t\epsilon}$
- Choose $t = \ln(1/(\delta\varphi))/\epsilon$ to get desired probability bound
- Space: $E[\,|S|\,] \leq n \times (t/n) = t$ (each stream item creates new entry in S with probability $\leq t/n$). Space independent of n and constant for constant ϕ, ϵ, δ!

Dealing with unknown n

Let $t = \ln(1/(\delta\varphi))/\epsilon$ be defined as before. We apply the same algorithm, with sampling rate adjusted dynamically to size of the stream seen so far to make sure that sampling probability is at least t/n:

- First $2t$ items are sampled with prob. 1
- For $i = 1, 2, \ldots$, next batch of $2^i t$ items sampled with prob. 2^{-i}
- At the beginning of each batch the sample S has to be recalibrated to reflect the new sampling rate as follows. For each $(x, f_e(x)) \in S$:
  - Let $t_x = \#$tails before head of unbiased coin
  - If $f_e(x) - t_x > 0$ then $f_e(x) = f_e(x) - t_x$
  - If $f_e(x) - t_x \leq 0$ then delete $(x, f_e(x))$ from S

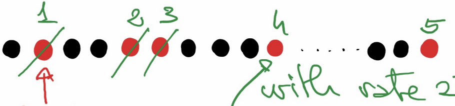
Dealing with unknown n (cont'd)

Remark

After the frequency adjustment, S is the same that would be obtained by applying the current sampling rate from the beginning

● : generic stream item

● : x Assume $t_x = 3$ (3 tails before head)



; $f_e(x) = f_e(x) - r$

first sample (rate $2^{-(i-1)}$)

with rate 2^{-i}:

Prob(first sampling at 4th occurrence)

$= (1/2)^3 (1 - 1/2)$

Dealing with unknown n (cont'd)

Lemma

Let $|\Sigma| = n$. At any time during the algorithm the sampling probability is at least t/n.

Proof.

Trivial when sampling with probability 1. For $i \geq 1$, when we start sampling with probability 2^{-i}, we have seen at least $2t + (\sum_{j=1}^{i-1} 2^j)t = 2^i t$ stream items. Therefore $n \geq 2^i t$, whence $2^{-i} \geq t/n$. Analogously, $2^{-i} \leq 2t/n$ $\square$

Remark

Since we sample with at least as frequently as in the case when n is known, algorithms correctly solves ϵ-AFI with prob. $\geq 1 - \delta$. Also, $E[S] \leq n \times (2t/n) \leq 2t$!

Sketches

A sketch is a space-efficient data structure that can be used to provide (usually randomized) estimates of (statistical) characteristics of a data stream.

Frequency Moments

Let $\Sigma = x_1, x_2, \ldots, x_n$ be drawn from a universe U of size u. Let f_u be the frequency of $u \in U$ in Σ: $f_u = |\{j : x_j = u, 1 \leq j \leq n\}|$).
For $k \geq 0$, the k-th frequency moment F_k of Σ is

$$F_k = \sum_{u \in U} f_u^k$$

- F_0 is the number of distinct items in Σ (letting $0^0 = 0$)
- $F_1 = n$
- F_2 is the Gini index of Σ (provides info on data skew)

Computing F_1 using small space is easy ($O(\log n)$-bit counter).
What about $k \neq 1$?

Streaming algorithm for computing F_0

- $\Sigma = x_1, x_2, \ldots \in U^\star$ ($|\Sigma|$ unknown). Observe that $F_0 \leq |U|$.
- To compute F_0 exactly: $F_0 = O(|U|)$ counters
- Probabilistic Counting: approximate F_0 using exponentially smaller space ($\log |U|$ bits) with probabilistic guarantees.

Probabilistic Counter

- Array C of $\log |U|$ bits
- Hash Function $h : U \rightarrow [0..|U| - 1]$. (Assume that h is fully random)
- Function $t(i)$: given $i \in [0..|U| - 1]$ returns number of trailing zeroes in binary representation of i. ($i = 12 = (1100)_2 \rightarrow t(i) = 2$)
- All elements of C are initialized to zero. Upon seeing x_i, set $C[t(h(x_i)] = 1$. When Σ ends, let R be the largest index of C with $C[R] = 1$. Return $\tilde{F}_0 = 2^R$.

Probabilistic guarantees (sketch, see [DF08])

- Intuition: If h is fully random, there will be on average $|U|/2^j$ values mapped to values vith at least j trailing zeroes ($|U|/2^j$ are the integers in $[0..|U|-1]$ with at least j trailing zeroes in their binary representation).
- In order to set the j-th most significant bit of C with constant probability, the stream must contain $\Omega(2^j)$ distinct items!

Lemma

If $Z_j = \#$ distinct items $x \in \Sigma$ with $t(h(x)) \geq j$, then $E[Z_j] = F_0/2^j$ and $Var[Z_j] < E[Z_j]$.

Proof.

Z_j can be seen as the sum of F_0 i.i.d. Bernoulli variables W_x, one for each distinct item $x \in \Sigma$, whose value is 1 if $t(h(x)) \geq j$. We have that $\Pr(W_x = 1) = (F_0/2^j)/F_0 = 1/2^j$. Then $E[Z_j] = F_0/2^j$ and $Var[Z_j] = E[Z_j](1 - 1/2^j) < E[Z_j]$. □

Probabilistic guarantees (cont'd)

Theorem

Let 2^R *be the returned value. Than, for any* $c > 2$,

$$\Pr(F_0/c \leq 2^R \leq cF_0) \geq 1 - 2/c$$

Proof idea: Straightforward combination of previous Lemma + Markov and Chebyshev's inequalities (see [DF08])

Exercise: High Probability Guarantees

Devise a simple technique to obtain the same guarantees of the Theorem with high probability $1 - 1/|U|^k$. The space requirement should increase from $\log_2 |U|$ bits to $O\left(\log^2 |U|\right)$ bits.
Hint: Keep several independent replicas of the counter and ...

References

For references to seminal work on streaming see:

C. Demetrescu and I. Finocchi. Chapter 8: Algorithms for Data Streams. In *Handbook of Applied Algorithms*, Wiley-IEEE Press, 2008. (pdf provided in Moodle)