Administrivia

- List of potential projects will be out by the end of the week
- If you have specific project ideas, catch me during office hours (right after class) or email me to set up a meeting
- Short project proposals (1-2 pages) due in class 3/22 (Thursday after Spring break)
- Final project papers due in late May
- Midterm date – first week of April(?)

Data Stream Processing (Part II)

- Alon,, Matias, Szegedy. "The space complexity of approximating the frequency moments", ACM STOC'1996.
- Alon, Gibbons, Matias, Szegedy. "Tracking Join and Self-join Sizes in Limited Storage", ACM PODS'1999.
- *SURVEY-1:* S. Muthukrishnan. "Data Streams: Algorithms and Applications"
- *SURVEY-2:* Babcock et al. "Models and Issues in Data Stream Systems", ACM PODS'2002.

Overview

- Introduction & Motivation
- Data Streaming Models & Basic Mathematical Tools
- Summarization/Sketching Tools for Streams
  - Sampling
  - Linear-Projection (aka AMS) Sketches
    - *Applications:* Join/Multi-Join Queries, Wavelets
  - Hash (aka FM) Sketches
    - *Applications:* Distinct Values, Set Expressions

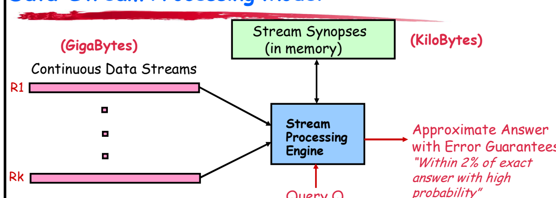
The Streaming Model

- Underlying signal: One-dimensional array A[1…N] with values A[i] all initially zero
  - Multi-dimensional arrays as well (e.g., row-major)
- Signal is implicitly represented via a stream of updates
  - j-th update is <k, c[j]> implying
    - A[k] := A[k] + c[j] (c[j] can be >0, <0)
- Goal: Compute functions on A[] subject to
  - Small space
  - Fast processing of updates
  - Fast function computation
  - …

Streaming Model: Special Cases

- Time-Series Model
  - Only j-th update updates A[j] (i.e., A[j] := c[j])
- Cash-Register Model
  - c[j] is always >= 0 (i.e., increment-only)
  - Typically, c[j]=1, so we see a multi-set of items in one pass
- Turnstile Model
  - Most general streaming model
  - c[j] can be >0 or <0 (i.e., increment or decrement)
- *Problem difficulty varies depending on the model*
  - E.g., MIN/MAX in Time-Series vs. Turnstile!

Data-Stream Processing Model



- Approximate answers often suffice, e.g., trend analysis, anomaly detection
- Requirements for stream synopses
  - *Single Pass:* Each record is examined at most once, in (fixed) arrival order
  - *Small Space:* Log or polylog in data stream size
  - *Real-time:* Per-record processing time (to maintain synopses) must be low
  - *Delete-Proof:* Can handle record deletions as well as insertions
  - *Composable:* Built in a *distributed fashion* and combined later

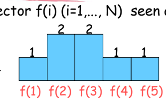
Probabilistic Guarantees

- Example: Actual answer is within 5 ± 1 with prob ≥ 0.9
- Randomized algorithms: Answer returned is a specially-built random variable
- User-tunable **(ε,δ)–approximations**
  - Estimate is within a relative error of ε with probability >= $1-\delta$
- Use Tail Inequalities to give probabilistic bounds on returned answer
  - Markov Inequality
  - Chebyshev's Inequality
  - Chernoff Bound
  - Hoeffding Bound

Linear-Projection (aka AMS) Sketch Synopses

- *Goal:* Build small-space summary for distribution vector f(i) (i=1,..., N) seen as a stream of i-values

Data stream: 3, 1, 2, 4, 2, 3, 5, . . .



- *Basic Construct:* *Randomized Linear Projection of f()* = project onto inner/dot product of f-vector

$$< f,\xi >= \sum f(i)\xi_i$$ where ξ = vector of random values from an appropriate distribution

  - Simple to compute over the stream: Add ξ_i whenever the i-th value is seen

Data stream: 3, 1, 2, 4, 2, 3, 5, . . . → $\xi_1 + 2\xi_2 + 2\xi_3 + \xi_4 + \xi_5$

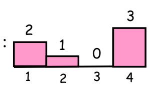
  - Generate ξ_i 's in small (logN) space using pseudo-random generators
  - *Tunable probabilistic guarantees* on approximation error
  - *Delete-Proof:* Just subtract ξ_i to delete an i-th value occurrence
  - *Composable:* Simply *add* independently-built projections

Example: Binary-Join COUNT Query

- Problem: Compute answer for the query COUNT(R $\bowtie_A$ S)
- Example:

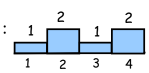
Data stream R.A: 4 1 2 4 1 4

$f_R(i)$:



Data stream S.A: 3 1 2 4 2 4

$f_S(i)$:



$$\text{COUNT}(R \bowtie_A S) = \sum_i f_R(i) \cdot f_S(i)$$

= 10 (2 + 2 + 0 + 6)

- Exact solution: too expensive, requires O(N) space!
  - N = sizeof(domain(A))

Basic AMS Sketching Technique [AMS96]

- Key Intuition: Use randomized linear projections of f() to define random variable X such that
  - X is easily computed over the stream (in small space)
  - E[X] = COUNT(R $\bowtie_A$ S)
  - Var[X] is small

→ Probabilistic error guarantees (e.g., actual answer is 10±1 with probability 0.9)

- Basic Idea:
  - Define a family of 4-wise independent {-1, +1} random variables

$$\{\xi_i : i = 1,\ldots,N\}$$

  - Pr[ξ_i = +1] = Pr[ξ_i = -1] = 1/2
    - Expected value of each ξ_i, E[ξ_i] = 0
  - Variables ξ_i are 4-wise independent
    - Expected value of product of 4 distinct ξ_i = 0
  - Variables ξ_i can be generated using pseudo-random generator using only O(log N) space (for seeding)!

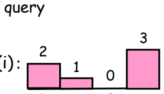
AMS Sketch Construction

- Compute random variables: $X_R = \sum_i f_R(i)\xi_i$ and $X_S = \sum_i f_S(i)\xi_i$
  - Simply add ξ_i to X_R(X_S) whenever the i-th value is observed in the R.A (S.A) stream
- Define X = $X_R X_S$ to be estimate of COUNT query
- Example:

Data stream R.A: 4 1 2 4 1 4

$X_R = X_R + \xi_4$

$f_R(i)$:

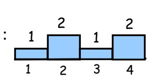


$X_R = 2\xi_1 + \xi_2 + 3\xi_4$

Data stream S.A: 3 1 2 4 2 4

$X_S = X_S + \xi_1$

$f_S(i)$:



$X_S = \xi_1 + 2\xi_2 + \xi_2 + 2\xi_4$

Binary-Join AMS Sketching Analysis

- Expected value of X = COUNT(R $\bowtie_A$ S)

$$E[X] = E[X_R \cdot X_S]$$
$$= E[\sum_i f_R(i)\xi_i \cdot \sum_i f_S(i)\xi_i]$$
$$= E[\sum_i f_R(i)\cdot f_S(i)\xi_i^2] + E[\sum_{i \neq i'} f_R(i)\cdot f_S(i')\xi_i\xi_{i'}]$$
$$= \sum_i f_R(i)\cdot f_S(i)$$

1

0

- Using 4-wise independence, possible to show that

$$\text{Var}[X] \leq 2 \cdot SJ(R) \cdot SJ(S)$$

- $SJ(R) = \sum_i f_R(i)^2$ is *self-join size of R (second/L2 moment)*

Boosting Accuracy

- Chebyshev's Inequality:

$$\Pr(|X - E[X]| \geq \varepsilon E[X]) \leq \frac{Var[X]}{\varepsilon^2 E[X]^2}$$

- Boost accuracy to ε by averaging over several independent copies of X (reduces variance)

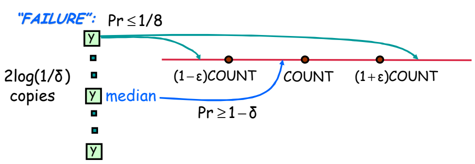
$$s = \frac{8 \cdot (2 \cdot SJ(R) \cdot SJ(S))}{\varepsilon^2 \, COUNT^2} \text{ copies} \qquad E[Y] = E[X] = COUNT(R \bowtie S)$$

- By Chebyshev: $Var[Y] = \frac{Var[X]}{s} \leq \frac{\varepsilon^2 \, COUNT^2}{8}$

$$\Pr(|Y - COUNT| \geq \varepsilon \cdot COUNT) \leq \frac{Var[Y]}{\varepsilon^2 \, COUNT^2} \leq \frac{1}{8}$$

Boosting Confidence

- Boost confidence to $1-\delta$ by taking median of $2\log(1/\delta)$ independent copies of Y
- Each Y = ***Bernoulli Trial***



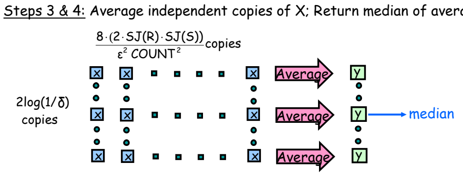
Pr[|median(Y)-COUNT| $\geq \varepsilon \cdot$ COUNT]

= Pr[# failures in $2\log(1/\delta)$ trials >= $\log(1/\delta)$]

$\leq \delta$ (By Chernoff Bound)

Summary of Binary-Join AMS Sketching

- Step 1: Compute random variables: $X_R = \sum_i f_R(i)\xi_i$ and $X_S = \sum_i f_S(i)\xi_i$
- Step 2: Define X= $X_R X_S$
- Steps 3 & 4: Average independent copies of X; Return median of averages



- Main Theorem (AGMS99): Sketching approximates COUNT to within a relative error of ε with probability $\geq 1-\delta$ using space

$$O\left(\frac{SJ(R) \cdot SJ(S) \cdot \log(1/\delta) \cdot \log N}{\varepsilon^2 \, COUNT^2}\right)$$

  - Remember: O(log N) space for "seeding" the construction of each X

A Special Case: Self-join Size

- Estimate COUNT(R $\bowtie_A$ R)= $\sum_i f_R^{\,2}(i)$ *(original AMS paper)*
- Second (L2) moment of data distribution, Gini index of heterogeneity, measure of skew in the data

In this case, COUNT = SJ(R), so we get an (ε,δ)–estimate using space only

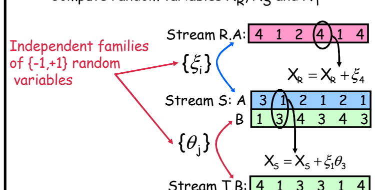
$$O\left(\frac{\log(1/\delta) \cdot \log N}{\varepsilon^2}\right)$$

Best-case for AMS streaming join-size estimation

What's the worst case??

AMS Sketching for Multi-Join Aggregates [DGGR02]

- Problem: Compute answer for COUNT(R $\bowtie_A$ S $\bowtie_B$ T) = $\sum_{i,j} f_R(i) f_S(i,j) f_T(j)$
- Sketch-based solution
  - Compute random variables X_R, X_S and X_T



$$X_R = \sum_i f_R(i)\xi_i = 2\xi_1 + \xi_2 + 3\xi_4$$

$$X_S = \sum_{i,j} f_S(i,j)\xi_i\theta_j = \xi_3\theta_1 + 3\xi_1\theta_3 + 2\xi_2\theta_4$$

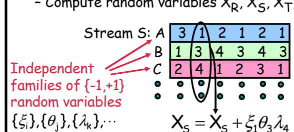
$$X_T = \sum_j f_T(j)\theta_j = 2\theta_1 + 2\theta_3 + 2\theta_4$$

  - Return X=$X_R X_S X_T$ (E[X]= COUNT(R $\bowtie_A$ S $\bowtie_B$ T))

$$E[f_R(i) \cdot f_S(i',j) \cdot f_T(j')\xi_i\xi_{i'}\theta_j\theta_{j'}] = 0 \quad \text{if } i \neq i' \text{ or } j \neq j'$$

AMS Sketching for Multi-Join Aggregates

- Sketches can be used to compute answers for general multi-join COUNT queries (over streams R, S, T, ...)
  - For each pair of attributes in equality join constraint, use independent family of {-1, +1} random variables
  - Compute random variables X_R, X_S, X_T, ...



$$X_S = \sum_{i,j,k\ldots} f_S(i,j,k,\cdots)\xi_i\theta_j\lambda_k \cdots$$

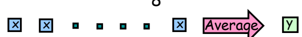
  - Return X=$X_R X_S X_T$ .... (E[X]= COUNT(R $\bowtie$ S $\bowtie$ T $\bowtie$ ....))

$$Var[X] \leq 2^{2m} \cdot SJ(R) \cdot SJ(S) \cdot SJ(T) \cdots$$

  - Explosive increase with the number of joins!

Boosting Accuracy by Sketch Partitioning: Basic Idea

- For ε error, need $\mathrm{Var}[Y] \le \dfrac{\varepsilon^2\ \mathrm{COUNT}^2}{8}$

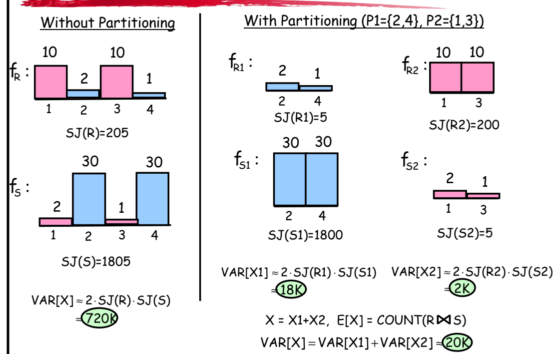


$$s = \frac{8 \cdot (2^{2m} SJ(R) \cdot SJ(S) \cdots)}{\varepsilon^2\ \mathrm{COUNT}^2} \quad \text{copies}$$

$$\mathrm{Var}[Y] = \frac{\mathrm{Var}[X]}{s} \le \frac{\varepsilon^2\ \mathrm{COUNT}^2}{8}$$

- Key Observation: Product of self-join sizes for partitions of streams can be *much smaller* than product of self-join sizes for streams
  - Reduce space requirements by partitioning join attribute domains
    - Overall join size = sum of join size estimates for partitions
  - Exploit coarse statistics (e.g., histograms) based on historical data or collected in an initial pass, to compute the *best partitioning*

Sketch Partitioning Example: Binary-Join COUNT Query



Overview of Sketch Partitioning

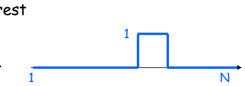
- Maintain independent sketches for partitions of join-attribute space
- Improved error guarantees
  - $\mathrm{Var}[X] = \sum \mathrm{Var}[Xi]$ is smaller (by *intelligent domain partitioning*)
  - *"Variance-aware" boosting:* More space to higher-variance partitions
- *Problem:* Given total sketching space S, find domain partitions p1,…, pk and space allotments s1,…,sk such that $\sum_j sj \le S$, and the variance

$$\frac{\mathrm{Var}[X1]}{s1} + \frac{\mathrm{Var}[X2]}{s2} + \cdots + \frac{\mathrm{Var}[Xk]}{sk} \quad \text{is minimized}$$

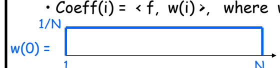
  - Solved optimal for binary-join case (using Dynamic-Programming)
  - NP-hard for ≥ 2 joins
    - Extension of our DP algorithm is an effective heuristic -- *optimal* for independent join attributes
- Significant accuracy benefits for small number (2-4) of partitions

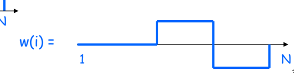
Other Applications of AMS Stream Sketching

- Key Observation: $|R1 \bowtie R2| = \sum f_1(i) f_2(i) = < f_1, f_2 >$ = *inner product!*
- *General result:* Streaming (ε, δ) estimation of "large" inner products using AMS sketching
- Other streaming inner products of interest
  - Top-k frequencies [CCF02]
    - Item frequency = < f, "unit_pulse" >



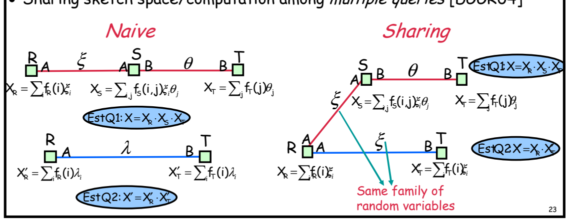
  - Large wavelet coefficients [GKMS01]
    - Coeff(i) = < f, w(i) >, where w(i) = i-th wavelet basis vector





More Recent Results on Stream Joins

- Better accuracy using *"skimmed sketches"* [GGR04]
  - "Skim" dense items (i.e., large frequencies) from the AMS sketches
  - Use the "skimmed" sketch only for sparse element representation
  - Stronger worst-case guarantees, and much better in practice
    - Same effect as sketch partitioning with *no apriori knowledge!*
- Sharing sketch space/computation among *multiple queries* [DGGR04]



Overview

- Introduction & Motivation
- Data Streaming Models & Basic Mathematical Tools
- Summarization/Sketching Tools for Streams
  - Sampling
  - Linear-Projection (aka AMS) Sketches
    - *Applications:* Join/Multi-Join Queries, Wavelets
  - Hash (aka FM) Sketches
    - *Applications:* Distinct Values, Set Expressions

Distinct Value Estimation

- Problem: Find the *number of distinct values* in a stream of values with domain [0,...,N-1]
  - Zeroth frequency moment F_0, L0 (Hamming) stream norm
  - Statistics: number of *species or classes* in a population
  - Important for query optimizers
  - *Network monitoring:* distinct destination IP addresses, source/destination pairs, requested URLs, etc.
- Example (N=64) Data stream: 3 0 5 3 0 1 7 5 1 0 3 7

  Number of distinct values: 5
- Hard problem for random sampling! [CCMN00]
  - Must sample almost the entire table to guarantee the estimate is within a factor of 10 with probability > 1/2, regardless of the estimator used!

Hash (aka FM) Sketches for Distinct Value Estimation [FM85]

- Assume a hash function h(x) that maps incoming values x in [0,..., N-1] *uniformly* across [0,..., 2^L-1], where L = O(logN)
- Let lsb(y) denote the position of the least-significant 1 bit in the binary representation of y
  - A value x is mapped to lsb(h(x))
- Maintain *Hash Sketch* = BITMAP array of L bits, initialized to 0
  - For each incoming value x, set BITMAP[lsb(h(x))] = 1

x = 5 ⟶ h(x) = 101100 ⟶ lsb(h(x)) = 2

BITMAP

5	4	3	2	1	0
0	0	0	1	0	0

Hash (aka FM) Sketches for Distinct Value Estimation [FM85]

- By uniformity through h(x): Prob[BITMAP[k]=1] = Prob[10^k] = $\frac{1}{2^{k+1}}$
  - Assuming d distinct values: expect d/2 to map to BITMAP[0], d/4 to map to BITMAP[1], . . .

BITMAP (L-1 … 0)

L-1																	0
0	0	0	0	0	0	1	0	1	0	1	1	1	1	1	1	1	1

position >> log(d) — fringe of 0/1s around log(d) — position << log(d)

- Let R = position of rightmost zero in BITMAP
  - Use as indicator of log(d)
- [FM85] prove that E[R] = $\log(\phi d)$, where $\phi = .7735$
  - Estimate d = $2^R/\phi$
  - Average several iid instances (different hash functions) to reduce estimator variance

Hash Sketches for Distinct Value Estimation

- [FM85] assume "ideal" hash functions h(x) (N-wise independence)
  - [AMS96]: pairwise independence is sufficient
    - h(x) = $(a \cdot x + b) \bmod N$, where a, b are random binary vectors in [0,...,2^L-1]
  - Small-space (ε, δ) estimates for distinct values proposed based on FM ideas
- *Delete-Proof:* Just use counters instead of bits in the sketch locations
  - +1 for inserts, -1 for deletes
- *Composable:* Component-wise OR/add distributed sketches together
  - Estimate |S1 ∪ S2∪…∪ Sk| = *set-union cardinality*