Data Stream Processing (Part I)

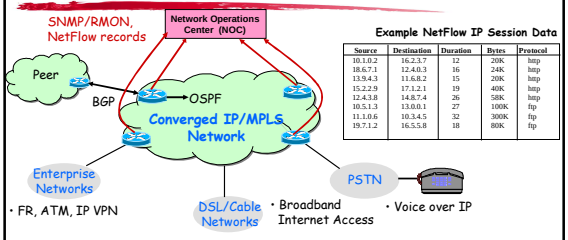
•Alon,, Matias, Szegedy. "The space complexity of approximating the frequency moments", *ACM STOC'1996.*
•Alon, Gibbons, Matias, Szegedy. "Tracking Join and Self-join Sizes in Limited Storage", *ACM PODS'1999.*
•*SURVEY-1:* S. Muthukrishnan. "Data Streams: Algorithms and Applications"
•*SURVEY-2:* Babcock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

Data-Stream Management

- Traditional DBMS – data stored in finite, persistent data sets
- Data Streams – distributed, continuous, unbounded, rapid, time varying, noisy, . . .
- Data-Stream Management – variety of modern applications
  - Network monitoring and traffic engineering
  - Telecom call-detail records
  - Network security
  - Financial applications
  - Sensor networks
  - Manufacturing processes
  - Web logs and clickstreams
  - Massive data sets

2

Networks Generate Massive Data Streams



Example NetFlow IP Session Data

Source	Destination	Duration	Bytes	Protocol
10.1.0.2	16.2.3.7	12	20K	http
18.6.7.1	12.4.0.3	16	24K	http
13.9.4.3	11.6.8.2	15	20K	http
15.2.2.9	17.1.2.1	19	40K	http
12.4.3.8	14.8.7.4	26	58K	http
10.5.1.3	13.0.0.1	27	100K	ftp
11.1.0.6	10.3.4.5	32	300K	ftp
19.7.1.2	16.5.5.8	18	80K	ftp

- SNMP/RMON/NetFlow data records arrive 24x7 from different parts of the network
- Truly massive streams arriving at rapid rates
  - AT&T collects 600-800 GigaBytes of NetFlow data each day!
- Typically shipped to a back-end data warehouse (off site) for off-line analysis

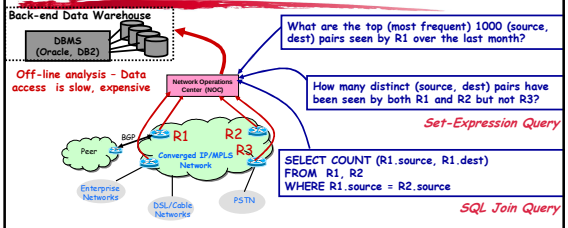
3

Packet-Level Data Streams

- Single 2Gb/sec link; say avg packet size is 50bytes
- Number of packets/sec = 5 million
- Time per packet = 0.2 microsec
- If we only capture header information per packet: src/dest IP, time, no. of bytes, etc. – at least 10bytes.
  - Space per second is 50Mb
  - Space per day is 4.5Tb per link
  - ISPs typically have hundred of links!
- Analyzing packet content streams – whole different ballgame!!

4

Real-Time Data-Stream Analysis



- Need ability to process/analyze network-data streams *in real-time*
  - As records stream in: look at records *only once in arrival order!*
  - Within resource (CPU, memory) limitations of the NOC
- Critical to important NM tasks
  - Detect and react to Fraud, Denial-of-Service attacks, SLA violations
  - Real-time traffic engineering to improve load-balancing and utilization

5

IP Network Data Processing

- Traffic estimation
  - How many bytes were sent between a pair of IP addresses?
  - What fraction network IP addresses are active?
  - List the top 100 IP addresses in terms of traffic
- Traffic analysis
  - What is the average duration of an IP session?
  - What is the median of the number of bytes in each IP session?
- Fraud detection
  - List all sessions that transmitted more than 1000 bytes
  - Identify all sessions whose duration was more than twice the normal
- Security/Denial of Service
  - List all IP addresses that have witnessed a sudden spike in traffic
  - Identify IP addresses involved in more than 1000 sessions

6

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
  - …

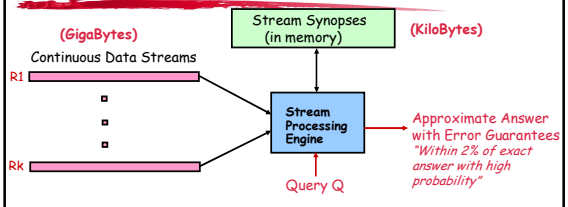
Example IP Network Signals

- Number of bytes (packets) sent by a source IP address during the day
  - 2^(32) sized one-d array; increment only
- Number of flows between a source-IP, destination-IP address pair during the day
  - 2^(64) sized two-d array; increment only, aggregate packets into flows
- Number of active flows per source-IP address
  - 2^(32) sized one-d array; increment and decrement

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

Data Stream Processing Algorithms

- Generally, algorithms compute approximate answers
  - Provably difficult to compute answers accurately with limited memory
- Approximate answers - Deterministic bounds
  - Algorithms only compute an approximate answer, but bounds on error
- Approximate answers - Probabilistic bounds
  - Algorithms compute an approximate answer with high probability
    - With probability at least $1-\delta$, the computed answer is within a factor ε of the actual answer
- Single-pass algorithms for processing streams also applicable to (massive) terabyte databases!

Sampling: Basics

- Idea: A small random sample S of the data often well-represents all the data
  - For a fast approx answer, apply "modified" query to S
  - Example: select agg from R where R.e is odd

    Data stream: 9 3 5 2 7 1 6 5 8 4 9 1 (n=12)

    Sample S: 9 5 1 8
  - If agg is avg, return average of odd elements in S — answer: 5
  - If agg is count, return average over all elements e in S of
    - n if e is odd — answer: 12*3/4 =9
    - 0 if e is even

Unbiased: For expressions involving count, sum, avg: the estimator is unbiased, i.e., the expected value of the answer is the actual answer

13

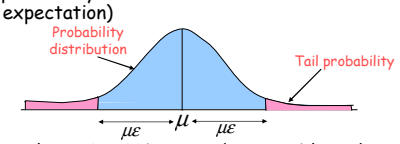
Probabilistic Guarantees

- Example: Actual answer is within 5 ± 1 with prob ≥ 0.9
- Randomized algorithms: Answer returned is a specially-built random variable
- Use Tail Inequalities to give probabilistic bounds on returned answer
  - Markov Inequality
  - Chebyshev's Inequality
  - Chernoff Bound
  - Hoeffding Bound

14

Basic Tools: Tail Inequalities

- General bounds on *tail probability* of a random variable (that is, probability that a random variable deviates far from its expectation)



- Basic Inequalities: Let X be a random variable with expectation μ and variance Var[X]. Then for any $\varepsilon > 0$

Markov: $\Pr(X \geq \varepsilon) \leq \frac{\mu}{\varepsilon}$

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

15

Tail Inequalities for Sums

- Possible to derive stronger bounds on tail probabilities for the sum of independent random variables
- Hoeffding's Inequality: Let X1, ..., Xm be independent random variables with 0<=Xi <= r. Let $\overline{X} = \frac{1}{m}\sum_i X_i$ and μ be the expectation of $\overline{X}$. Then, for any $\varepsilon > 0$,

$$\Pr(|\overline{X} - \mu| \geq \varepsilon) \leq 2\exp^{\frac{-2m\varepsilon^2}{r^2}}$$

- Application to avg queries:
  - m is size of subset of sample S satisfying predicate (3 in example)
  - r is range of element values in sample (8 in example)
- Application to count queries:
  - m is size of sample S (4 in example)
  - r is number of elements n in stream (12 in example)
- More details in [HHW97]

16

Tail Inequalities for Sums

- Possible to derive even stronger bounds on tail probabilities for the sum of independent *Bernoulli trials*
- Chernoff Bound: Let X1, ..., Xm be independent Bernoulli trials such that Pr[Xi=1] = p (Pr[Xi=0] = 1-p). Let $X = \sum_i X_i$ and $\mu = mp$ be the expectation of X. Then, for any $\varepsilon > 0$,

$$\Pr(|X - \mu| \geq \mu\varepsilon) \leq 2\exp^{\frac{-\mu\varepsilon^2}{2}}$$

- Application to count queries:
  - m is size of sample S (4 in example)
  - p is fraction of odd elements in stream (2/3 in example)
- Remark: Chernoff bound results in tighter bounds for count queries compared to Hoeffding's inequality

17

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

18

Computing Stream Sample

- Reservoir Sampling [Vit85]: Maintains a sample S of a fixed-size M
  - Add each new element to S with probability M/n, where n is the current number of stream elements
  - If add an element, evict a random element from S
  - Instead of flipping a coin for each element, determine the number of elements to skip before the next to be added to S
- Concise sampling [GM98]: Duplicates in sample S stored as <value, count> pairs (thus, potentially boosting actual sample size)
  - Add each new element to S with probability 1/T (simply increment count if element already in S)
  - If sample size exceeds M
    - Select new threshold T' > T
    - Evict each element (decrement count) from S with probability 1-T/T'
  - Add subsequent elements to S with probability 1/T'

19

Synopses for Relational Streams

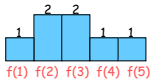
- Conventional data summaries fall short
  - Quantiles and 1-d histograms [MRL98,99], [GK01], [GKMS02]
    - Cannot capture attribute correlations
    - Little support for approximation guarantees
  - Samples (e.g., using Reservoir Sampling)
    - Perform poorly for joins [AGMS99] or distinct values [CCMN00]
    - Cannot handle deletion of records
  - Multi-d histograms/wavelets
    - Construction requires multiple passes over the data
- Different approach: *Pseudo-random sketch synopses*
  - Only logarithmic space
  - *Probabilistic guarantees* on the quality of the approximate answer
  - Support insertion as well as deletion of records *(i.e., Turnstile model)*

20

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

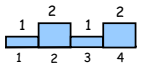
21

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

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



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

= 10 (2 + 2 + 0 + 6)

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

22

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)!

23

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:

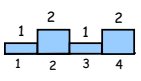
Data stream R.A: 4 1 2 (4) 1 4

$X_R = X_R + \xi_4$

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

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

$X_S = X_S + \xi_1$



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

24

Binary-Join AMS Sketching Analysis

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

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

ξ_i^2 → 1; $E[\sum_{i \neq i'} f_R(i) \cdot f_S(i')\xi_i\xi_{i'}]$ → 0

- Using 4-wise independence, possible to show that

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

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

25