Querying Distributed Data Streams

Minos Garofalakis
Technical University of Crete
Software Technology and Network Applications Lab

http://www.softnet.tuc.gr/~minos/

Work with: Haifa U, Technion, U Neuchatel, TU Dresden
Big Data is Big News (and Big Business...)

• Rapid growth due to several information-generating technologies, such as mobile computing, sensornets, and social networks

• How can we cost-effectively manage and analyze all this data...?
Big Data Challenges: The Four V’s – and one D

• **Volume:** Scaling from Terabytes to Exa/Zettabytes

• **Velocity:** Processing massive amounts of *streaming data*

• **Variety:** Managing the complexity of multiple relational and non-relational data types and schemas

• **Veracity:** Handling inherent uncertainty and noise in the data

• **Distribution:** Dealing with massively distributed information

• **Focus:** Volume, Velocity, Distribution
Velocity: *Continuous Stream Querying*

There are many scenarios where we need to monitor/track events over streaming data:

- Network health monitoring within a large ISP
- Collecting and monitoring environmental data with sensors
- Observing usage and abuse of large-scale data centers
Stream Processing Model

- Approximate answers often suffice, e.g., trends, anomalies
- Requirements for stream synopses
  - Single Pass: Each record examined at most once, in arrival order
  - Small Space: Log or polylog in data stream size
  - Small Time: Per-record processing time must be low
  - Also: Delete-proof, Composable, ...

![Diagram of Stream Processing Model](image)
Model of a Relational Stream

- Relation “signal”: **Large** array $v_S[1...N]$ with values $v_S[i]$ initially zero
  - Frequency-distribution array of $S$
  - Multi-dimensional arrays as well (e.g., row-major)

- Relation implicitly rendered via a **stream of updates**
  - Update $<x, c>$ implying
    - $v_S[x] := v_S[x] + c$ (c can be $>0$, $<0$)

- **Goal**: Compute queries (functions) on such dynamic vectors in “small” space and time ($<< N$)
Velocity & Distribution: Continuous Distributed Streaming

- Other structures possible (e.g., hierarchical, P2P)
- Goal: Continuously track (global) query over streams at coordinator
  - Using small space, time, and communication
  - Example queries:
    - Join aggregates, Variance, Entropy, Information Gain, ...
Example: LEADS Elastic μClouds Architecture

(http://leads-project.eu)
Continuous Distributed Streaming

- But... local site streams continuously change! New readings/data...
- Classes of monitoring problems
  - **Threshold Crossing**: Identify when $f(S) > \tau$
  - **Approximate Tracking**: $f(S)$ within guaranteed accuracy bound $\theta$
    - Tradeoff accuracy and communication / processing cost
- Naïve solutions must *continuously* centralize all data
  - Enormous communication overhead!
- Instead, *in-situ* stream processing using local constraints!

Monitor $f(S_1,...,S_m)$

S_1


S_m
Key Idea: "Push-based" in-situ processing
- Local filters installed at sites process local streaming updates
  - Offer bounds on local-stream behavior (at coordinator)
  - "Push" information to coordinator only when filter is violated
  - "Safe"! Coordinator sets/adjusts local filters to guarantee accuracy

- Easy for linear functions! Exploit additivity...
- Non-linear $f()$ ...??
Outline

- Introduction: Continuous Distributed Streaming
- The Geometric Method (GM)
- Recent Work: GM + Sketches, GM + Prediction Models
- Future Directions & Conclusions
Monitoring General, Non-linear Functions

Query: $f(S_1, ..., S_k) > \tau$ ?

- For general, non-linear $f()$, problem is a lot harder!
  - E.g., information gain over global data distribution
- Non-trivial to decompose the global threshold into “safe” local site constraints
  - E.g., consider $N = (N_1 + N_2)/2$ and $f(N) = 6N - N^2 > 1$
    Tricky to break into thresholds for $f(N_1)$ and $f(N_2)$
The Geometric Method

• A general purpose geometric approach [SKS SIGMOD’06]
  – Monitor function domain rather than the range of values!

• Each site tracks a local statistics vector $v_i$ (e.g., data distribution)

• Global condition is $f(v) > \tau$, where $v = \sum_i \lambda_i \ v_i$ ($\sum_i \lambda_i = 1$)
  – E.g., $v = \text{average}$ of local statistics vectors

• All sites share estimate $e = \sum_i \lambda_i \ v_i'$ of $v$
  based on latest update $v_i'$ from site $i$

• Each site $i$ tracks its drift from its most recent update $\Delta v_i = v_i - v_i'$
Covering the Convex Hull

- Key observation: \( \mathbf{v} = \sum_i \lambda_i \cdot (\mathbf{e} + \Delta \mathbf{v}_i) \)
  (a convex combination of “translated” local drifts)

- \( \mathbf{v} \) lies in the convex hull of the \((\mathbf{e} + \Delta \mathbf{v}_i)\) vectors

- Convex hull is completely covered by spheres with radii \( ||\Delta \mathbf{v}_i/2||_2 \) centered at \( \mathbf{e} + \Delta \mathbf{v}_i/2 \)

- Each such sphere can be constructed independently
Monochromatic Regions

• **Monochromatic Region:** For all points \( x \) in the region \( f(x) \) is on the same side of the threshold (\( f(x) > \tau \) or \( f(x) \leq \tau \))

• Each site independently checks its sphere is monochromatic
  – Find max and min for \( f() \) in local sphere region (may be costly)
  – Send updated value of \( v_i \) if not monochrome
Restoring Monochromicity

\[ f(x) > \tau \]
Restoring Monochromaticity

• After update, $||\Delta v_i||_2 = 0 \Rightarrow$ Sphere at $i$ is monochromatic
  – Global estimate $e$ is updated, may cause more site updates

• Coordinator case: Can allocate local slack vectors to sites to enable “localized” resolutions
  – Drift (=radius) depends on slack (adjusted locally for subsets)
Extensions: Transforms, Shifts, Safe Zones

• Subsequent developments [SKS TKDE’12]
  – Extend spheres to more general ellipsoids
  – Different reference vectors can be used to increase distance from threshold
  – Combining these observations allows additional cost savings

• More general theory of “Safe Zones”
  – Convex subsets of the admissible region where drift vectors may lie
  – SZs can be fine-tuned based on the function and give provably better performance
    • Recent work (under revision for VLDB)...
Outline

• Introduction: Continuous Distributed Streaming

• The Geometric Method (GM)

• Recent Work: GM + Sketches, GM + Prediction Models

• Future Directions & Conclusions
Continuous approximate monitoring
- Maintain the value of a function to within specified accuracy bound $\theta$

Too much local info $\Rightarrow$ Local summaries at sites
- A form of dimensionality reduction
- Bounding regions for the lower-dimensional sketching space
- Function over sketch $\Rightarrow$ Sketching error $\varepsilon$
  - Accounted for in the threshold checks (depend on both $\varepsilon, \theta$)

Key Problems:
1. Minimize data exchange volume
2. Deal with highly-nonlinear AMS estimator
AMS Sketches 101

- Simple randomized linear projections of data distribution
  - Easily computed over stream using logarithmic space
  - **Linear**: Compose through simple vector addition

\[
\text{\textbf{AMS Sketches 101}}
\]

\[
\{\xi_i\} \quad \{\psi_i\}
\]

\[
X_i = \sum_i v[i] \xi_i = \xi_1 + 2\xi_2 + 2\xi_3 + \xi_4 + \xi_5
\]

\[
X_K = \sum_i v[i] \psi_i
\]

\[
\text{sk}(v) = 
\]

• Simple randomized linear projections of data distribution
  – Easily computed over stream using logarithmic space
  – **Linear**: Compose through simple vector addition
Tracking Complex Aggregate Queries

- **Class of queries**: Generalized inner products of streams

\[ |R \Join S| = f_R \cdot f_S = \sum_v f_R[v] f_S[v] \]

- Join/multi-join aggregates, range queries, heavy hitters, histograms, wavelets, ...
Monitored Function…?

**AMS Estimator function for Self-Join**

\[
f(sk(v)) = \text{median}_{i=1..n} \left\{ \frac{1}{m} \sum_{j=1}^{m} sk(v)[i, j]^2 \right\} = \text{median}_{i=1..n} \left\{ \frac{1}{m} \| sk(v)[i] \|^2 \right\}
\]

- **Theorem (AMS96):** Sketching approximates \( \| v \|_2^2 \) to within an error of \( \pm \varepsilon \| v \|_2^2 \) with probability \( \geq 1 - \delta \) using \( O\left(\frac{1}{\varepsilon^2 \log(1/\delta)}\right) \) counters.
Geometric Query Monitoring using AMS Sketches [GKS VLDB’13]

• Deciding ball monochromicity for the median
  – Fast greedy algorithm for determining the distance to the inadmissible region

• (Non-trivial) extension to general inner product (join) queries

• Minimizing volume of data exchanges
  – Sketches can still get pretty large!
  – Can reduce problem to monitoring in $O(\log(1/\delta))$ dimensions
Outline

• Introduction: Continuous Distributed Streaming

• The Geometric Method (GM)

• Recent Work: GM + Sketches, GM + Prediction Models

• Future Directions & Conclusions
Exploiting Shared Prediction Models

• Naïve "static" prediction: Local stream assumed “unchanged” since last update
  – No update from site ⇒ last update (“predicted” value) is unchanged ⇒ global estimate vector unchanged

• *Dynamic prediction models* of site behavior
  – Built locally at sites and *shared* with coordinator
  – Model complex stream patterns, reduce number of updates
  – But... more complex to maintain and communicate
## Adapting Local Prediction Models

### [CG VLDB’05, TODS’08]

<table>
<thead>
<tr>
<th>Model</th>
<th>Predicted $v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Growth</td>
<td>$v_i^p(t) = \frac{t}{t_s} v_i(t_s)$</td>
</tr>
<tr>
<td>Velocity/ Acceleration</td>
<td>$v_i^p(t) = v_i(t_s) + (t - t_s) vel_i + (t - t_s)^2 acc_i$</td>
</tr>
<tr>
<td>Static</td>
<td>Equivalent to the basic framework</td>
</tr>
<tr>
<td></td>
<td>$v_i^p(t) = v_i(t_s)$</td>
</tr>
</tbody>
</table>

Predicted Global Vector: $$e^p(t)=\sum \lambda_i v_i^p(t)$$
Prediction-based Geometric Monitoring

[GDG SIGMOD’12, TODS’14]

\[ f(v(t)) > T \]

- Stricter local constraints if local predictions remain accurate
- Keep up with \( v(t) \) movement
Outline

• Introduction: Continuous Distributed Streaming

• The Geometric Method (GM)

• Recent Work: GM + Sketches, GM + Prediction Models

• Future Directions & Conclusions
Work in CD Streaming

• Much interest in these problems in TCS and DB areas

• Many functions of (global) data distribution studied:
  – Set expressions [Das,Ganguly,G,Rastogi’04]
  – Quantiles and heavy hitters [Cormode,G, Muthukrishnan, Rastogi’05]
  – Number of distinct elements [Cormode et al.,’06]
  – Spectral properties of data matrix [Huang,G, et al.’06]
  – Anomaly detection in networks [Huang ,G, et al.’07]
  – Samples [Cormode et al.’10]
  – Counts, frequencies, ranks [Yi et al.,’12]

• See proceedings of recent NII Shonan meeting on Large-Scale Distributed Computation
  http://www.nii.ac.jp/shonan/seminar011/
Monitoring Systems

- Much theory developed, but less progress on deployment
- Some empirical study in the lab, with recorded data
- Still applications abound: Online Games [Heffner, Malecha’09]
  - Need to monitor many varying stats and bound communication
  - Also, Distributed CEP systems (FERARI project)

- Several steps to follow:
  - Build libraries of code for basic monitoring problems
  - Evolve these into general purpose systems (distributed DBMSs?)

- Several questions to resolve:
  - What functions to support? General purpose, or specific?
  - What keywords belong in a query language for monitoring?
CD Monitoring in Scalable Network Architectures

- E.g., DHT-based P2P networks

- Single query point
  - “Unfolding” the network gives hierarchy
  - But, single point of failure (i.e., root)

- Decentralized monitoring
  - Everyone participates in computation, all get the result
  - Exploit epidemics? Latency might be problematic...
Theoretical Foundations

“Communication complexity” studies lower bounds of distributed one-shot computations

• Gives lower bounds for various problems, e.g., count distinct (via reduction to abstract problems)

• Need new theory for continuous computations
  – Based on info. theory and models of how streams evolve?
  – Link to distributed source coding or network coding?

Slepian-Wolf theorem [Slepian Wolf 1973]

http://www.networkcoding.info/
https://buffy.eecs.berkeley.edu/PHP/resabs/resabs.php?f_year=2005&f_submit=chapgrp&f_chapter=1
Challenges, challenges, challenges...

• Distributed streaming versions of hard analytics functions (e.g., PageRank)?

• Guaranteeing privacy of sensitive data in μClouds?

• Geometric monitoring for Distributed CEP hierarchies?
  – Deal with uncertain events (“V” for Veracity)?

• Implementing GM ideas in scalable stream-processing engines (e.g., Storm)?

• CD machine learning to dynamically adapt to data/workload conditions?
  – Communication just one of our concerns

• Scalable, adaptive analytics tools for massive, streaming time series?
Conclusions

- Continuous querying of distributed streams is a natural model
  - Interesting space/time/communication tradeoffs
  - Captures several real-world applications

- **Geometric Method**: Generic tool for monitoring complex, non-linear queries
  - Sketches [GKS VLDB’13], dynamic prediction models [GDG SIGMOD’12, TODS’14], Skyline Monitoring [PG ICDE’14]

- Much non-trivial algorithmic and theoretical work in CDS model
  - Intense research interest from DB and TCS communities
  - Deployment in real systems to come...

- **Much interesting work to be done!**
PS. We are hiring... 😊

FET Flagship (2013- …)
http://humanbrainproject.eu

ICT STREP (2012-5)
http://leads-project.eu

Flexible Event Processing for Big Data Architectures
ICT STREP (2014-7)
http://ferari-project.eu

Configurable, Autonomously-Adaptive Real-time Data Processing
ICT STREP (2014-7)
http://qualimaster.eu
Thank you!

http://www.softnet.tuc.gr/~minos/
minos@softnet.tuc.gr