Monitoring Distributed Streams using Convex Decompositions

Arnon Lazerson, Izchak Sharfman, Assaf Schuster - Technion I.I.T.
Daniel Keren - Haifa University
Minos Garofalakis, Vasilis Samoladas - Technical University of Crete.

Acknowledgments: This work was supported by the European Commission under ICT-FP7-FERARI-619491 (Flexible Event Processing for big Data Architectures)

Background

Large-scale monitoring applications rely on continuous tracking of complex queries over distributed data streams.

- Effective distributed stream processing solutions must be
  - Space efficient
  - Time efficient
  - **Communication efficient**

Why Distributed Stream Monitoring?

Two highly important research areas:

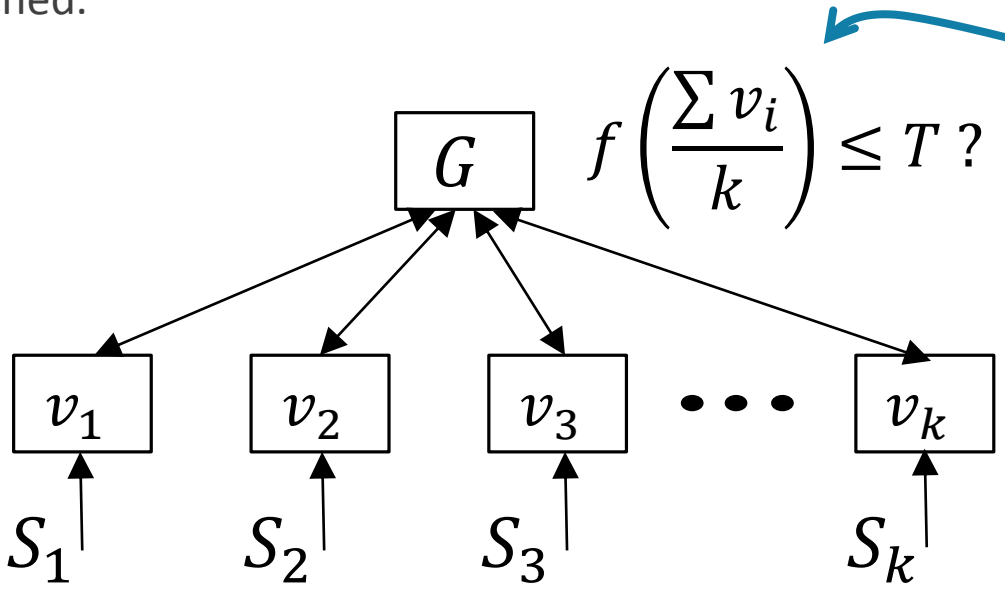
- **Stream analysis**
- **Distributed optimization/model fitting/learning…**

BUT

- **Suppose you've solved a distributed problem – and then the data at the nodes begins to change.**
- **It's impossible to either centralize the data or re-run the distributed algorithm every time this happens.**
- **Practical compromise: define *LOCAL* conditions at the nodes, such that as long as they hold, it is guaranteed than the *GLOBAL* solution did not change by more than a given amount.**

Distributed Monitoring Model

- Distributed streams S_i continuously update the local vectors v_i at the nodes.
- The remote nodes communicate with a designated coordinator site G.
- The coordinator G must issue an alert when the global condition $f\left(\frac{\sum v_i}{k}\right) \leq T$ is breached.



A rather general model, which describes many practically important problems. Can be further enriched by augmenting the local vectors v_i with functions of their raw coordinates.

It's not enough to check that every node satisfies the condition…

- Two nodes hold (local) contingency tables
- The global contingency table is the average of the local ones.
- Given a table π, the mutual information M is defined as

$$M(\pi) = \pi_{11} \log\left[\frac{\pi_{11}}{(\pi_{11}+\pi_{12})(\pi_{11}+\pi_{21})}\right] + \pi_{12} \log\left[\frac{\pi_{12}}{(\pi_{11}+\pi_{12})(\pi_{12}+\pi_{22})}\right] + \ldots$$

- We wish to know when $M(\,) > T$

It's not enough to check that every node satisfies the condition…

- Say we wish to know when $M(\,) > 0.4$
- Assume for example:

$$\pi = \begin{pmatrix} 0.9 & 0.03 \\ 0.02 & 0.05 \end{pmatrix},\quad \pi' = \begin{pmatrix} 0.04 & 0.02 \\ 0.03 & 0.91 \end{pmatrix}:\ M(\pi) = 0.104, M(\pi') = 0.082,$$

- Note that each node satisfies the condition
- However the global condition is violated:

$$M\left(\frac{\pi + \pi'}{2}\right) = 0.494$$

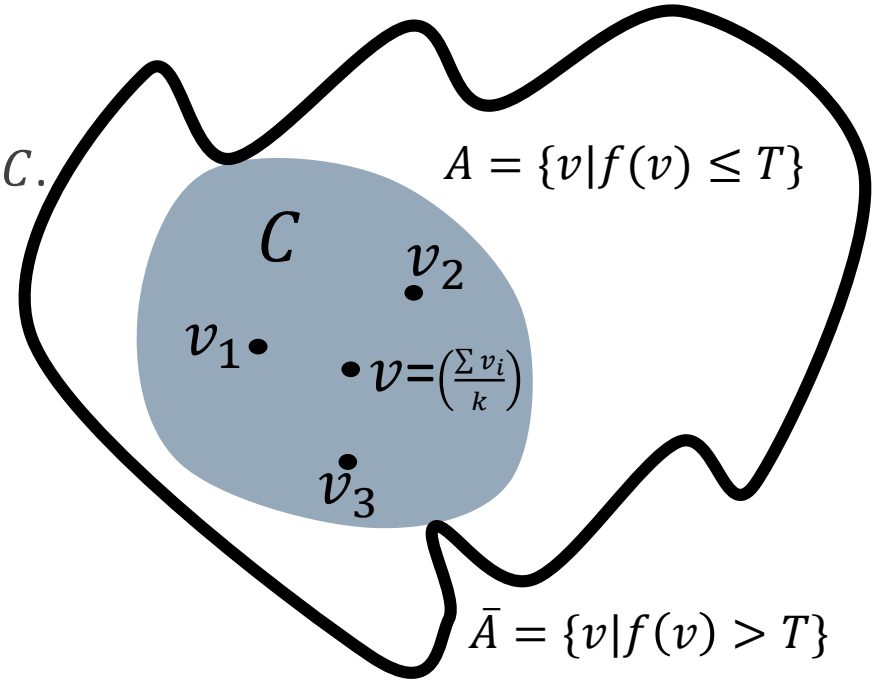
- M of the global table is much larger than M of each local value.
- **There is no way to infer about the global M given the local ones**.

Practical Implication

- We a have a distributed mail-server and we wish to know if a specific word is indicative of spam.
- A word may not be indicative of spam when looking at the local contingency tables, but it is indicative of spam when taking the global (correct) table.
- The opposite is also true.
- This does NOT happen for linear aggregates, counting problems etc.

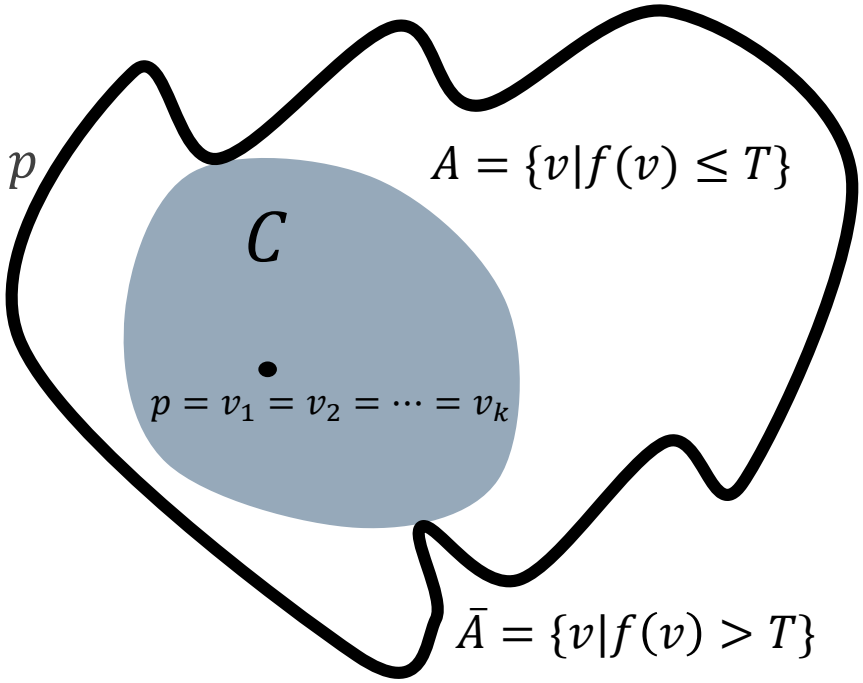
Geometric Monitoring

- The set $A = \{v | f(v) \leq T\}$ is called the *admissible region.*
- C is a *convex* subset of A.
- C's convexity guarantees that if $v_1, v_2 \ldots v_k \in C$, then $v = \left(\frac{\sum v_i}{k}\right) \in C \subseteq A$.
- Node i can stay silent as long as $v_i \in C$.



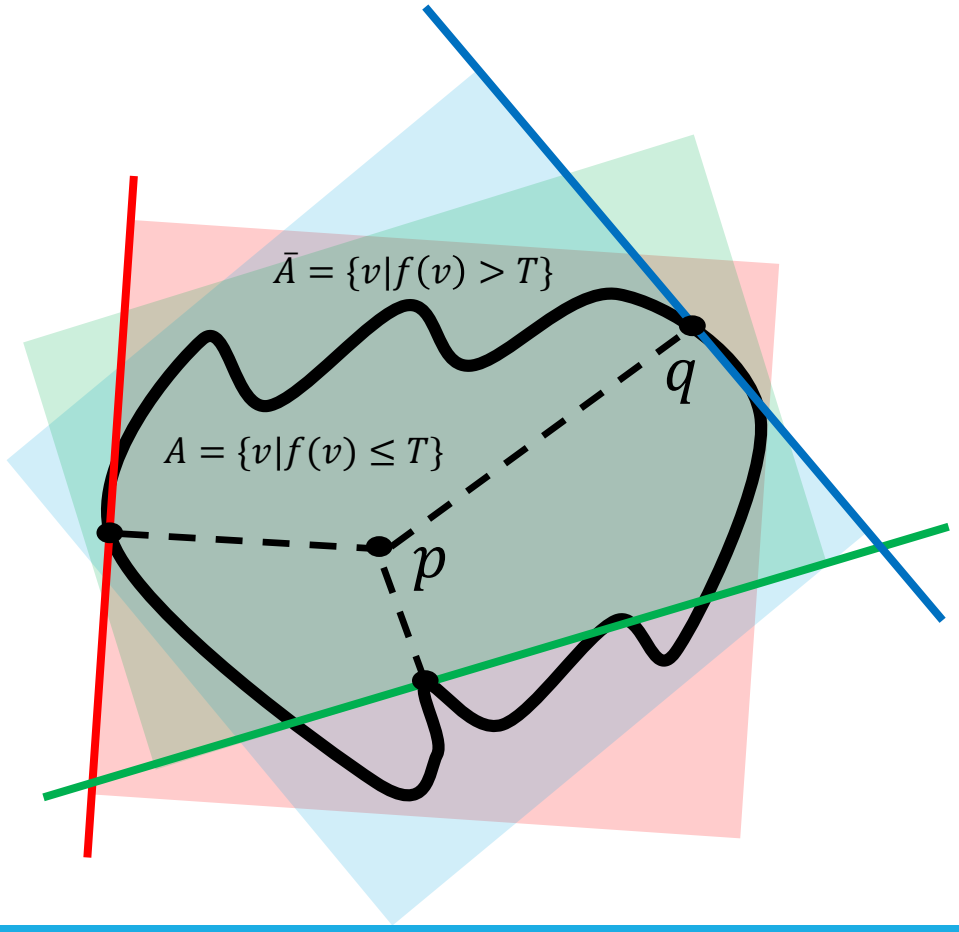
Properties of C

- For simplicity assume that initially $v_1 = v_2 = \cdots = v_k = p$.
- C must be *convex.*
- C must be included in A.
- C must include p.
- The boundary of C should be far from p

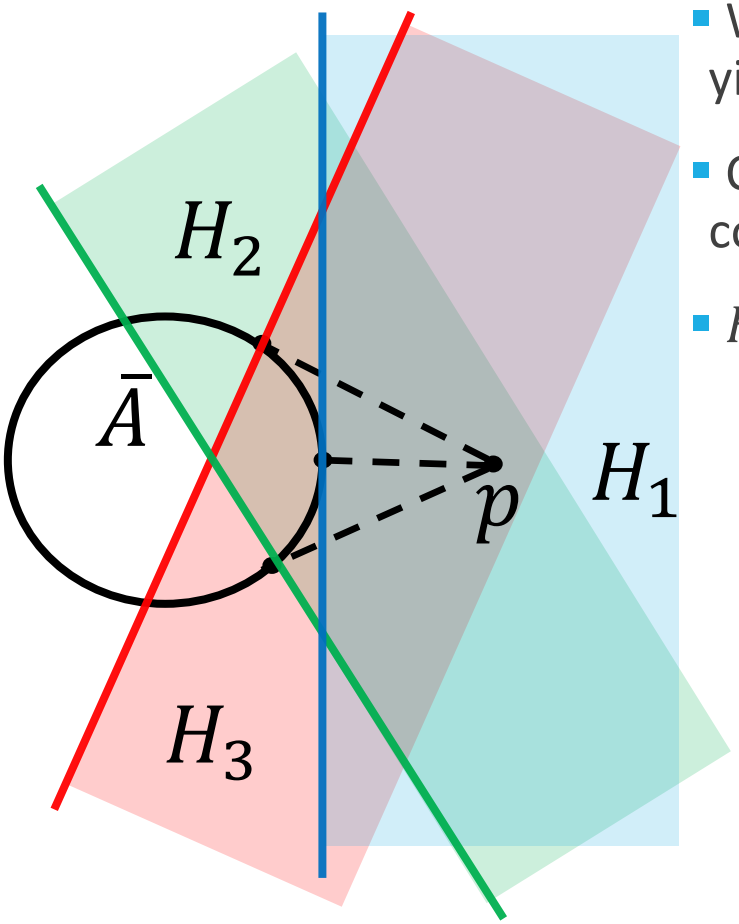


Previous Work (VLDB13 and earlier): Using half-spaces to construct C.

- We want to construct C a convex subset of A containing the point p
- C can be created using half-spaces:
  - For each point q on the boundary of $\bar{A}$ take the half-space through q orthogonal to the segment pq and containing p.
  - Then take the intersection of these half-spaces.
- Note 1: this is true in any dimension.
- Note 2: there is a way to define this set without actually intersecting all half-spaces.



Redundant half-spaces



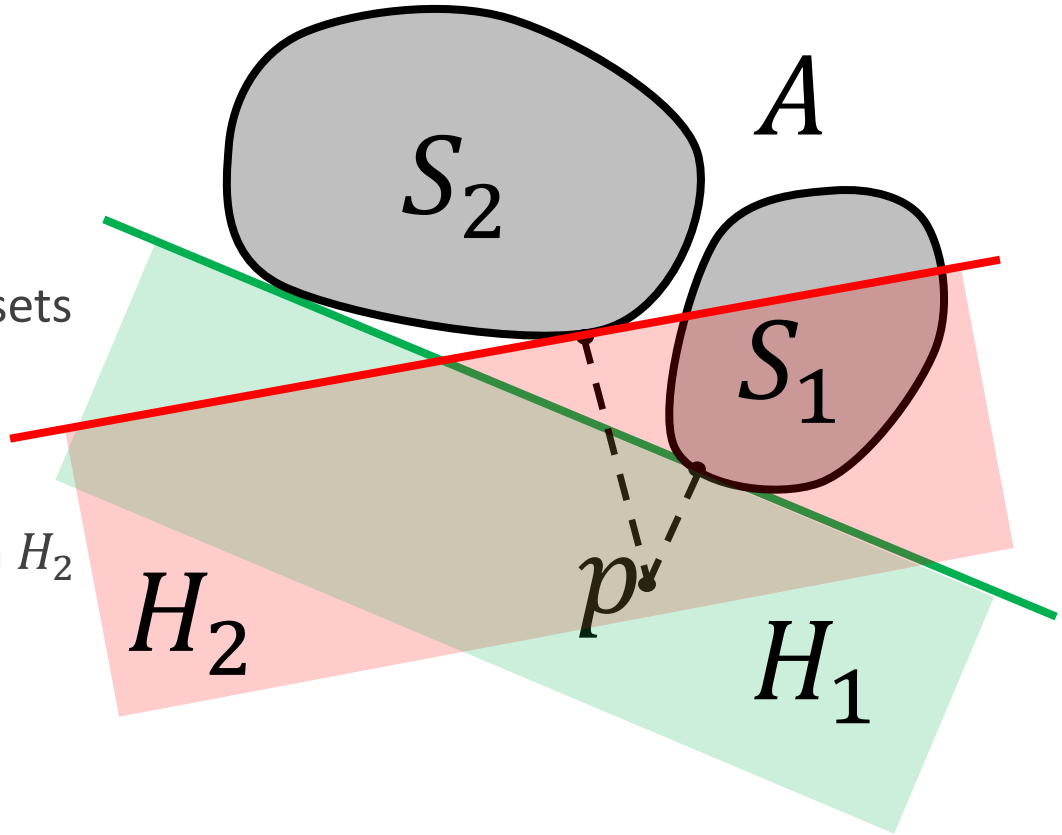
- When $\bar{A}$ is convex the previous method yields a non-optimal C.
- Clearly H_1 is a better convex subset (it contains $H_1 \cap H_2 \cap H_3$).
- H_2, H_3 are **"redundant".**

Convex Decomposition

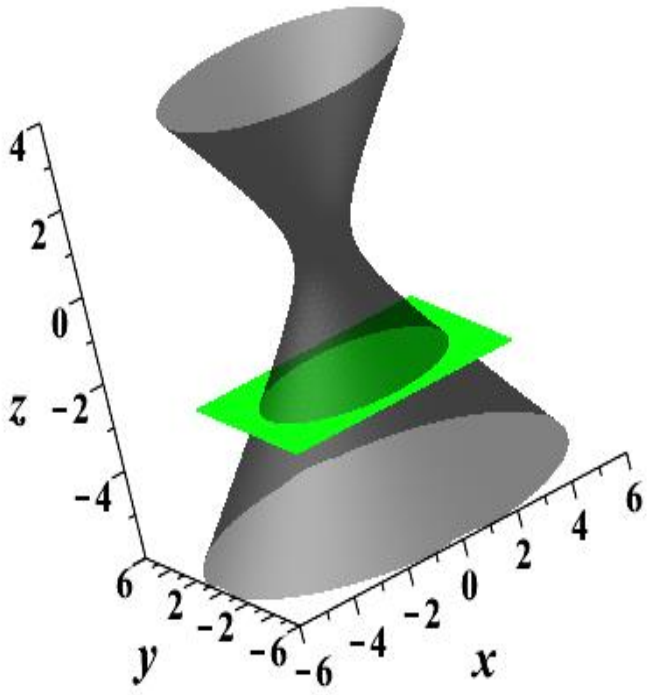
General case: ***convex decomposition*** of $\bar{A}$ but avoid redundancy!

Example:

- $\bar{A}$ is the union of the two convex sets S_1 and S_2
- H_1 separates p from S_1 and S_2
- There is no need to intersect with H_2
- C can be defined as H_1



Convex Decomposition



Another example

- $\bar{A}$ is the hyperboloid defined by $x^2 + y^2 - z^2 \leq 1$.
- Note that $\bar{A}$ is not convex.
- But it can be decomposed into a **infinite** union of (convex) parallel discs.
- It is **impossible** to decompose it into a finite union of convex subsets – yet, the proposed method can still be applied.
- This is a (very) special case of monitoring inner product with *AGMS sketches* (more later).

Example: Monitoring the median

- We wish to monitor $med(x_1, x_2, x_3) > 0$
- A is not convex.
  - The vectors $(1,1,-11)$, $(1,-11,1)$, $(-11,1,1)$ are in A, but their average $(-3,-3,-3)$ is not.
- We partition the complement $\bar{A}$ to convex subsets:
  $(x_1, x_2 \leq 0)$, $(x_1, x_3 \leq 0)$, $(x_2, x_3 \leq 0)$
- This is a non-redundant decomposition (can be proved).
- It can be generalized to any dimension.

* The number of sets in the decomposition is exponential, but a simple trick allows to check whether a point belongs to the A in $nlog(n)$ time, where n is the dimension.

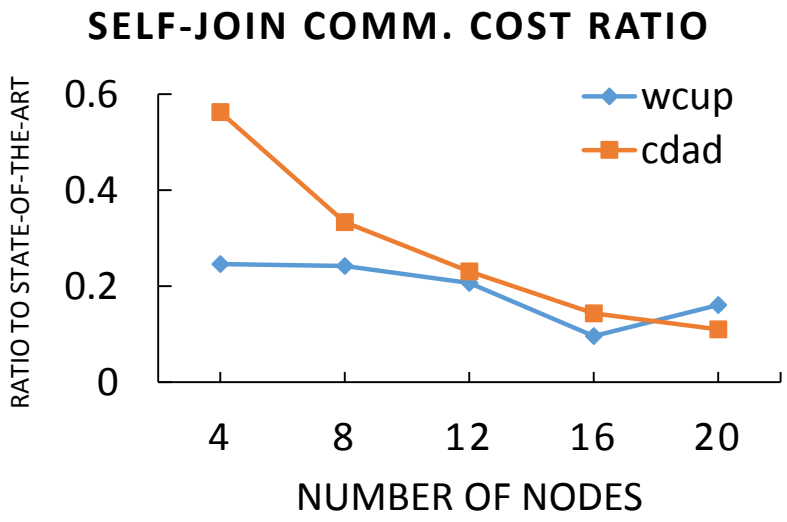
Application: AGMS Sketches

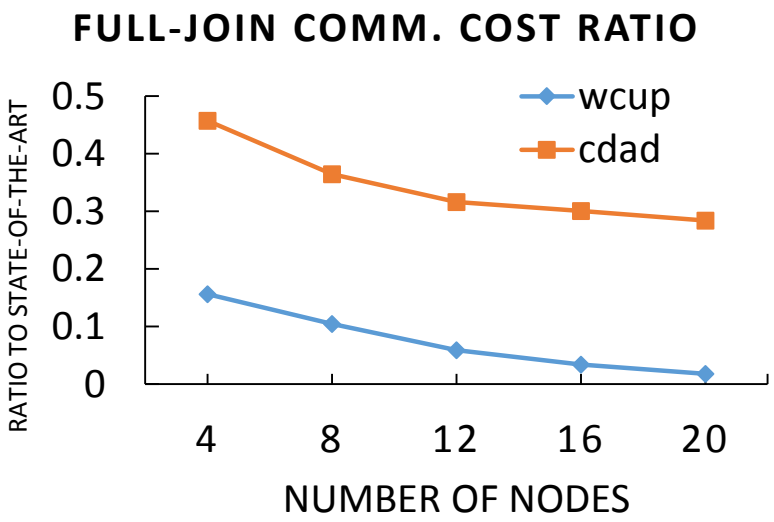
- AGMS sketches provide summary of the data using a $k \times l$ matrix M.
- They can be used to approximate the result of range, self-join and full-join queries
- To do so the median operator is applied to a vector constructed either by taking the norm squared of each matrix row, or by taking the inner-product of the rows of two sketch matrices
- We are dealing with median of a quadric function.
- This is more complex than the linear median case, but can be solved.

Experimental Results

- The CD method was evaluated using two real data sets:
  - "Wcup", which contains access logs to the soccer World Cup 1998 website.
  - "Cdad", which contains SNMP requests of network users, such as number of packets and bytes from/to each user's machine
- CD was applied both to self-join and full-join queries using sketches.
- It yielded substantial improvement over the state-of-the-art.

Experimental Results





Communication cost (bytes sent) relative to state-of-the-art for varying number of nodes over two datasets.

Thank you! Questions

Computing self-join in distributed setting

- N_i are the distinct nodes, each of which holds a matrix.
- The matrices in the distinct nodes are first averaged.
- Then the norm squared of each row in the average matrix is computed.
- Finally a median is taken over the resulting values.