Quality of Service Aware Data Stream Processing for Highly Dynamic and Scalable Applications

Isam Mashhour Hasan Al Jawarneh,

PhD Candidate, Cycle XXXII

Department of Computer Science and Engineering - DISI
University of Bologna, Italy
isam.aljawarneh3@unibo.it

Supervisor:
Prof. **Rebecca Montanari**

PhD program coordinator:
Prof. **Davide Sangiorgi**

Esame finale anno 2020
2nd April 2020

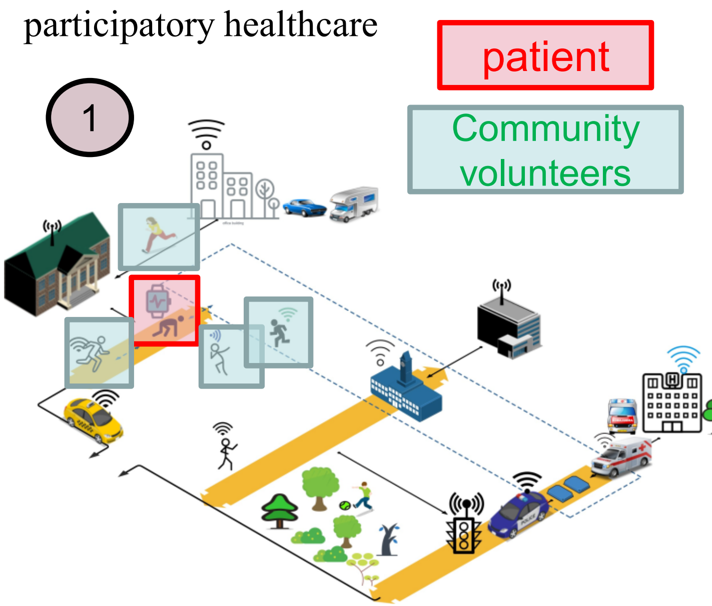
Outline

- **Motivation and thesis statement**
- Thesis contribution: a novel architecture for QoS-aware distributed processing of big spatial data
  - ➢ Processing and storage of static data
    - ✓ QoS-aware spatial **batch processing** engine
    - ✓ QoS-aware scalable **storage** for spatial data
  - ➢ Online Spatial Approximate Query Processing
    - ✓ QoS-aware **stream processing** engine for spatial data
    - ✓ Adaptive spatial **stream-static join** processor
- Conclusions and future works

Motivating Application Scenario

A mixed-workload scenario requiring at least:

- **Traffic Light Controller**. Actuator decides to change lights consistently for ambulance to pass
- **Smart Real-time Pathfinder**. **I**nteractive navigation map for ambulances and other vehicles
- **Real-time Community Detector**. Identify volunteers' communities in the surroundings of the patient



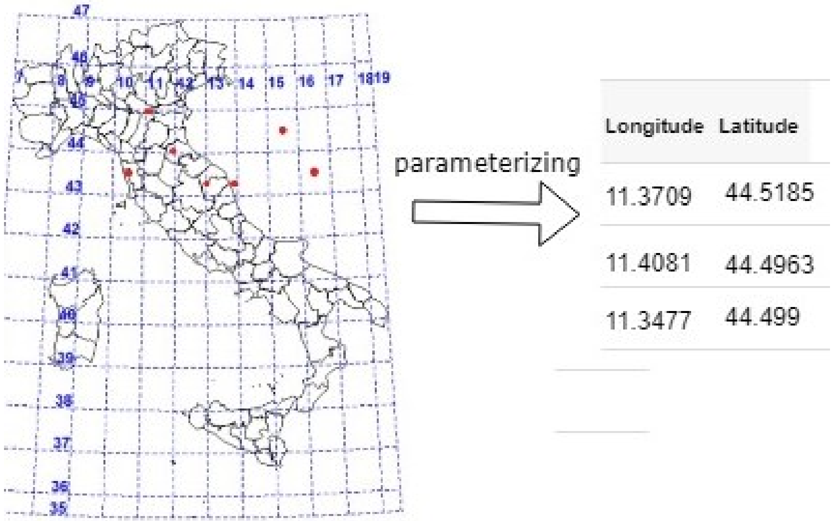
➢ **Primitive geospatial queries**

- Proximity queries
- Spatial join
- Spatial clustering
- Spatial geo-statistics.
- *k*-Nearest Neighborhoods)

Geospatial data representation

The problem of multidimensionality

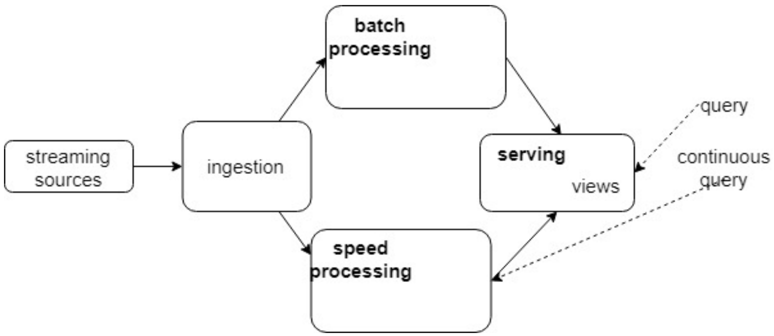
- Big geospatial data management is challenging
- A spatial point is parametrized and represented as coordinates (longitude and latitude)
- GPS is rarely 100% accurate, susceptible to acceptable error-bounds
- **Geometry** inherent in the data will be **lost** by such a transformation
- Spatial reconstruction is **expensive**



- Dimensionality reduction. **Geohash** is a geospatial encoding
- Generates a single-dimensional representation as a string that encompasses a geographical meaning

Related works

- Few ad-hoc fixes, patches and glues for spatial data management:
  - Spatial partitioning methods for **batch** processing
  - Spatial query **optimizers**
  - Spatial **stream** processing
- They do not collectively form a comprehensive architecture for mix workload demanding dynamic applications
- Drawbacks of Lambda architecture
  - Not attuned to the spatial characteristics of data
  - QoS-awareness is not incorporated transparently
  - Users need to explicitly reason about underlying logistics



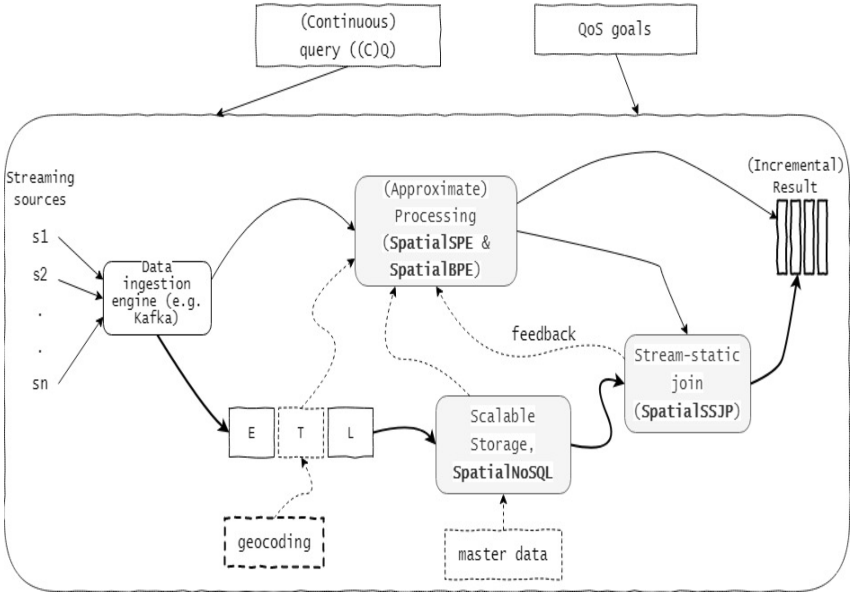
***Lambda* [1]**

[1] N. Marz and J. Warren, *Big Data: Principles and Best Practices of Scalable Real-Time Data Systems.* New York; Manning Publications Co., 2015.

SpatialDSMS: a platform for geo big data

- Design guidelines
  - A platform operating atop best-in-breed Cloud computing representatives
  - Same multidimensionality **reduction** approach for all workloads: storage, batch and stream processing
  - All layers collaborate **synergistically** in serving a mixed workload dynamic application scenario with QoS guarantees
  - Design goals include **modularity** and **composability**
  - Our new layer becomes the new codebase for QoS awareness

Spatial data stream management system

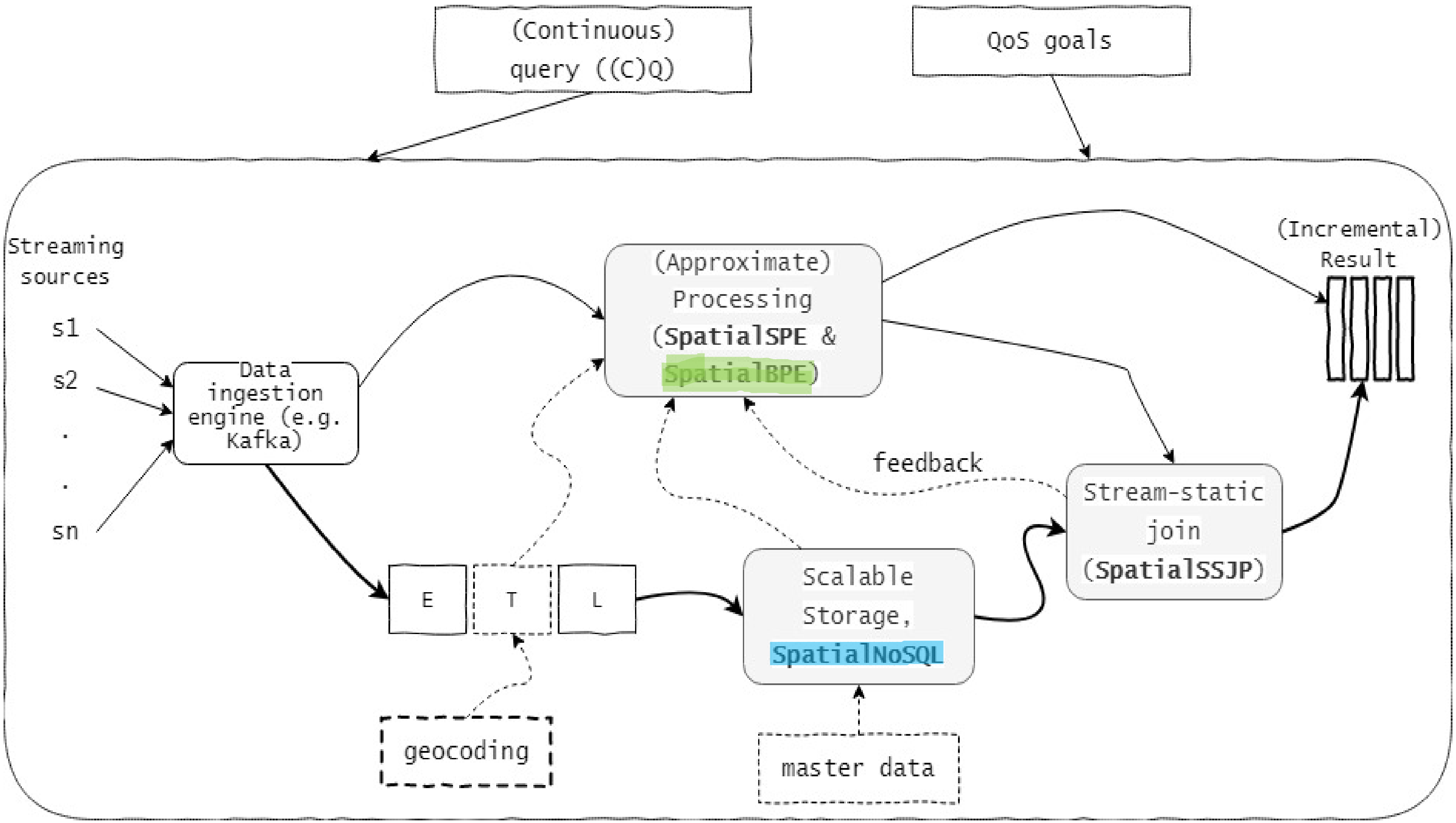


- **QoS** goals include
  - ✓ Low latency
  - ✓ High throughput
  - ✓ Maximum resource utilization
  - ✓ Maximum accuracy

Processing and storage of static data: optimization pillars

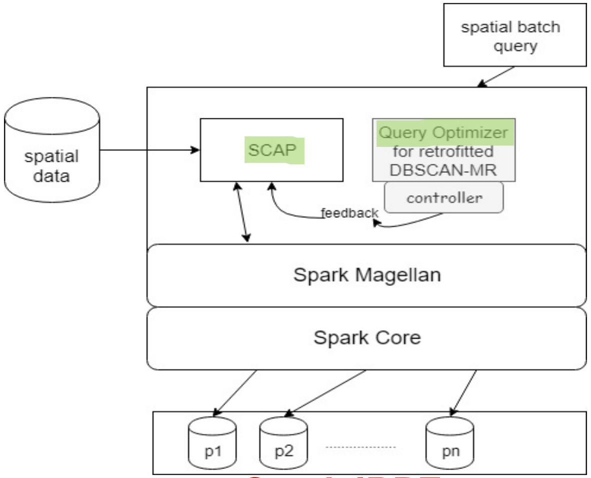
- Two aspects are important for optimizing the processing and storage of spatial data-at-rest:
  - Spatial **data partitioning**
    - ✓ Splitting data to parallelly internetworked processing (or storage) worker nodes
    - ✓ Sharding in NoSQL scalable storage is analogous
  - Spatial **query optimizers**
    - ✓ Optimizing spatial query by selecting the best performing query plan
- Spatial data partitioning goals
  - Load balancing
  - Spatial Data Locality (SDL) preservation
  - Boundary Spatial Objects (BSO) minimization

QoS-aware static big spatial data processing

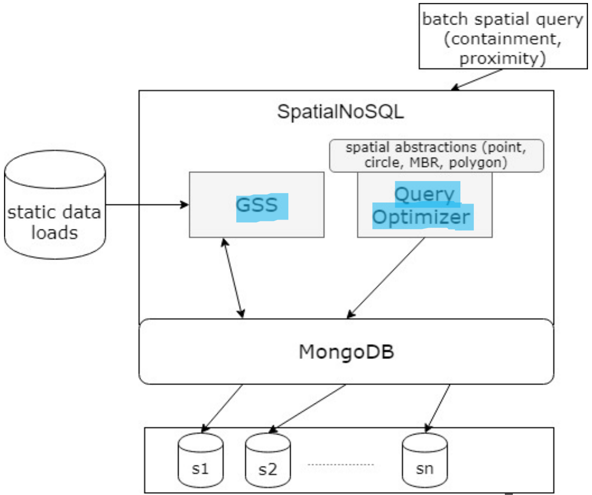


SpatialBPE and SpatialNoSQL: Overview

- Spatial QoS-aware batch processing system
- Two component:
  - Custom **partitioning method:** SCAP for batch systems and GSS for NoSQL storage
  - Spatial **query optimizer**, exploiting SCAP and GSS for clustering, spatial join, proximity and containment queries



SpatialBPE

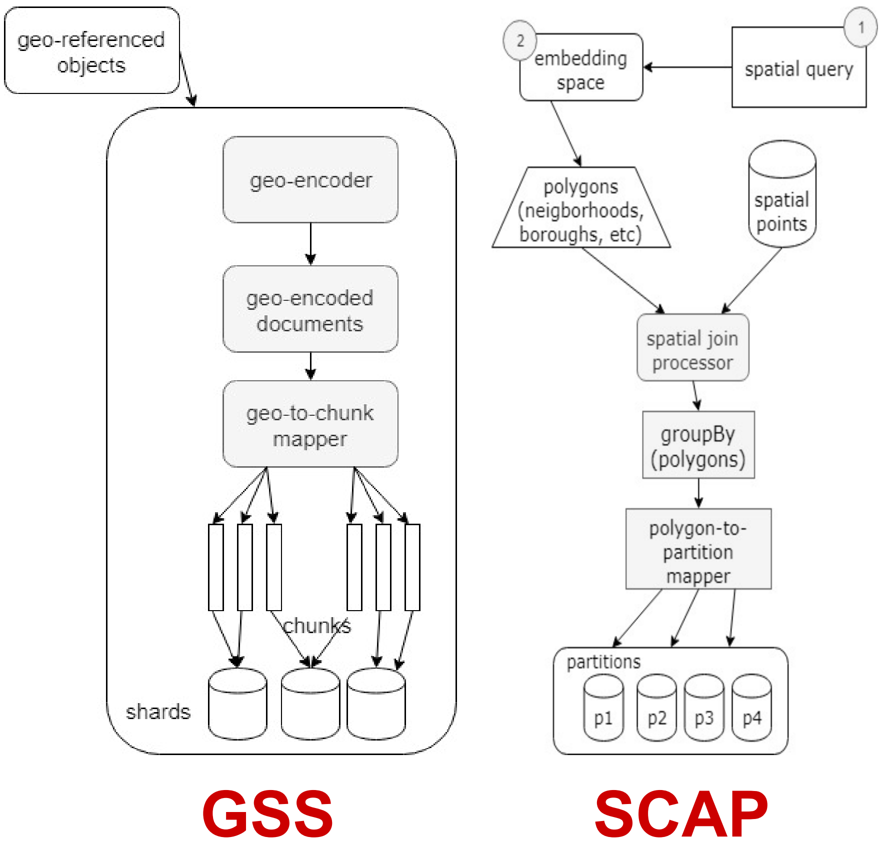


SpatialNoSQ

Spatial Co-locality-aware Partitioner (SCAP) & Geospatial Sharding Scheme (GSS)

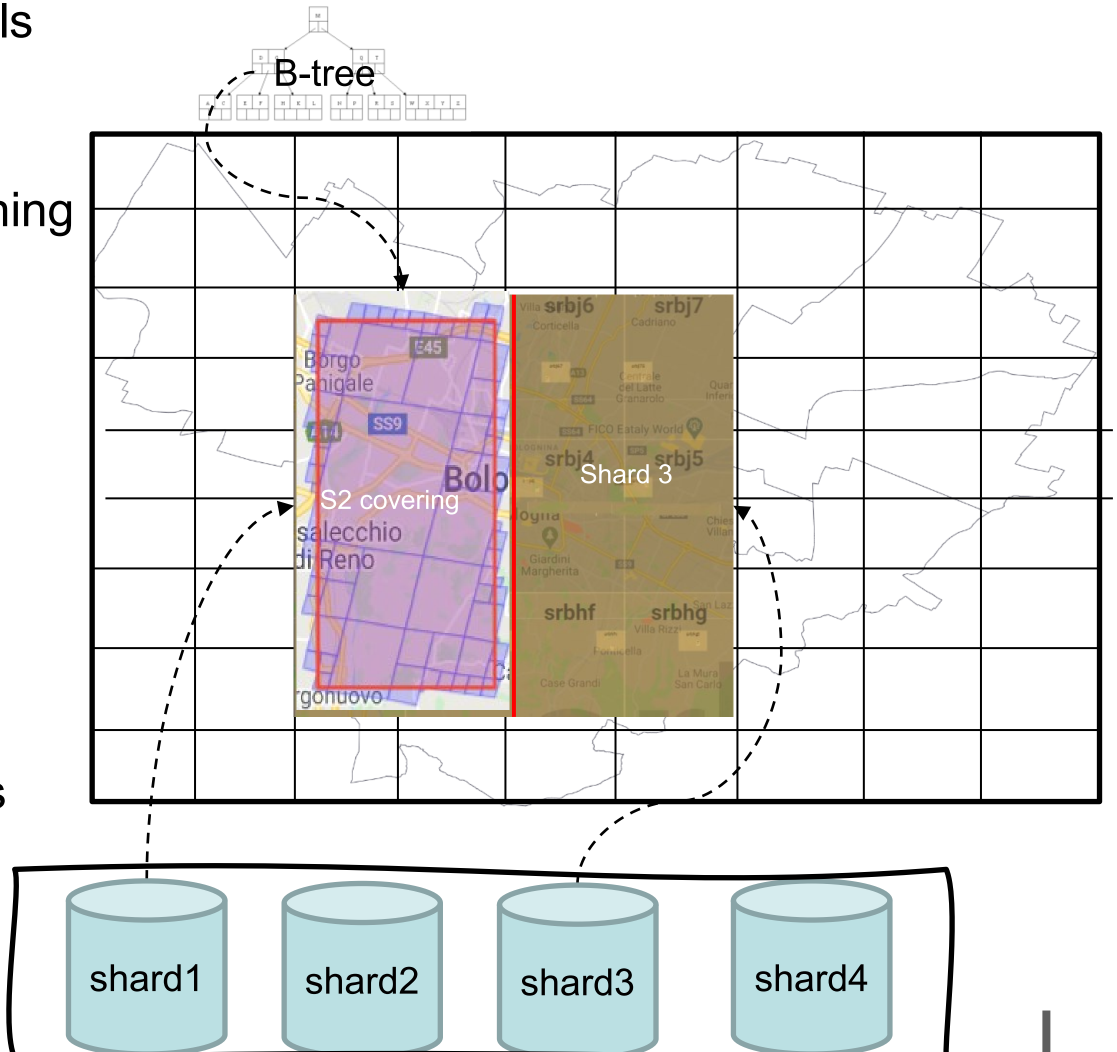
- **SDL** preservation is a priority
- **BSOs** and **load balancing** to a lesser extent

- ***Clump*** geometrically co-located objects into single chunks
- ***Split*** *overloaded* chunks
- ***Map*** chunks *to partitions*



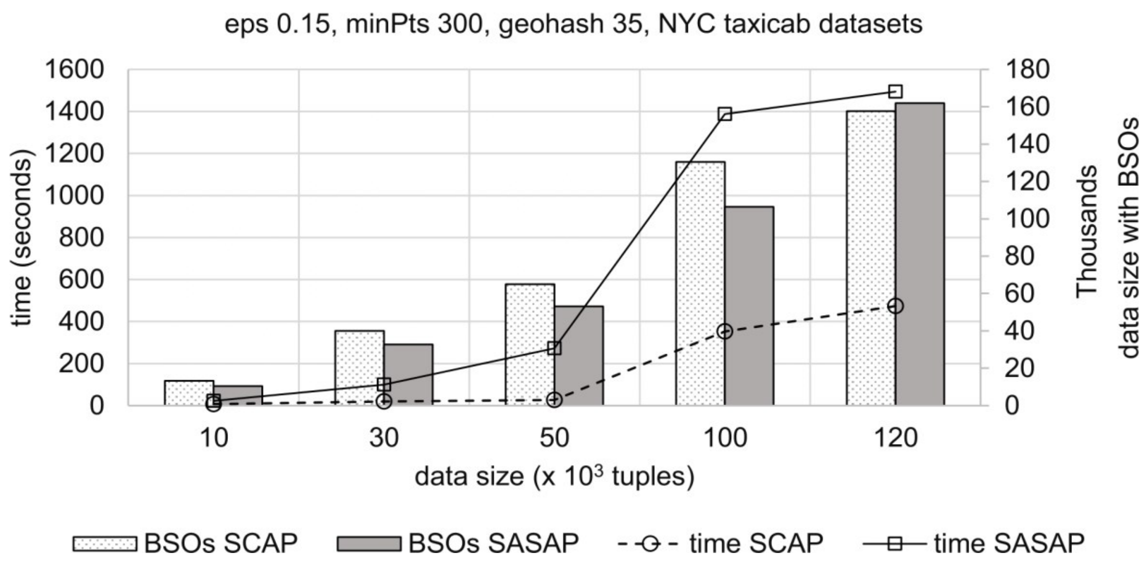
Spatial query optimizer for NoSQL: Overview

- We have created a two-levels indexing scheme
- MongoDB router forwards requests to few shards, pruning the search space
- **Overlay** the embedding space with a fixed-grid network
- **Generate** a geohash covering and a list of interacting points
- **Generate** S2 sub-coverings and retrieve interacting points
- **Impose** B-tree index on sub-coverings



Results: achieving QoS goals for clustering

Lower better

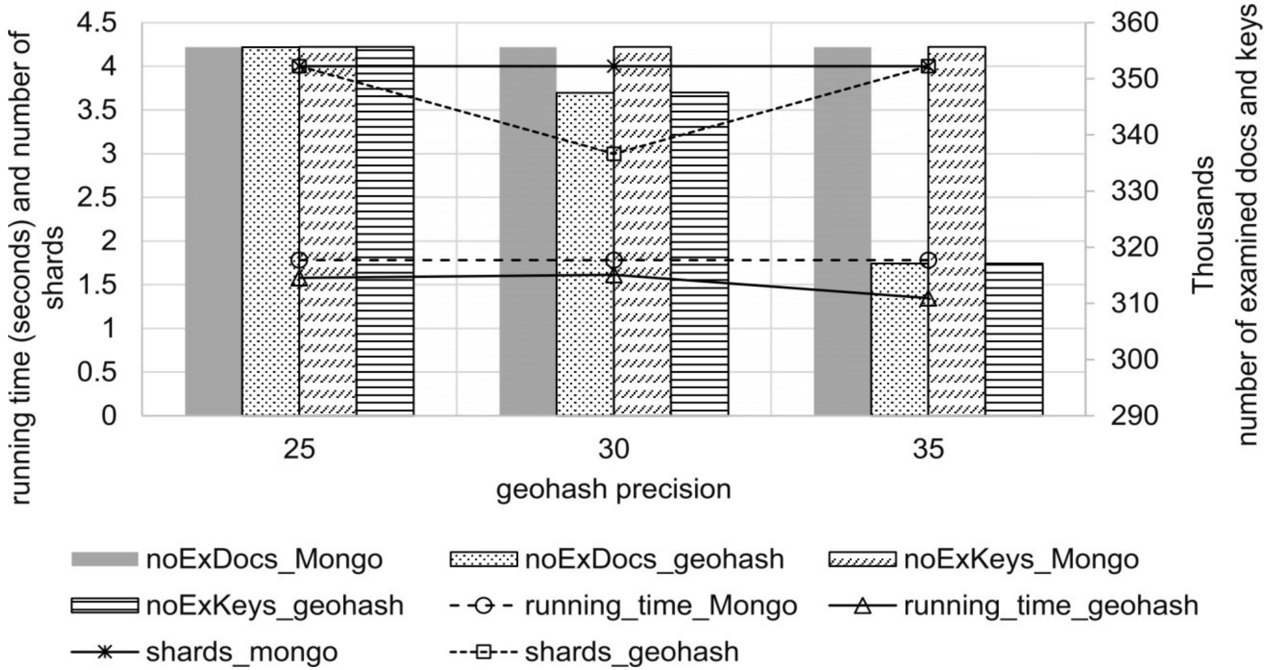


Lower better

- **NY City** taxicab itinerary datasets a cohort of approximately 150k
- 150k spatial data points that were collected through the **ParticipAct** project by our group in Bologna, Italy
- Tweak geohash from 30 to 35
- Total running time has been decreased

Results: containment test

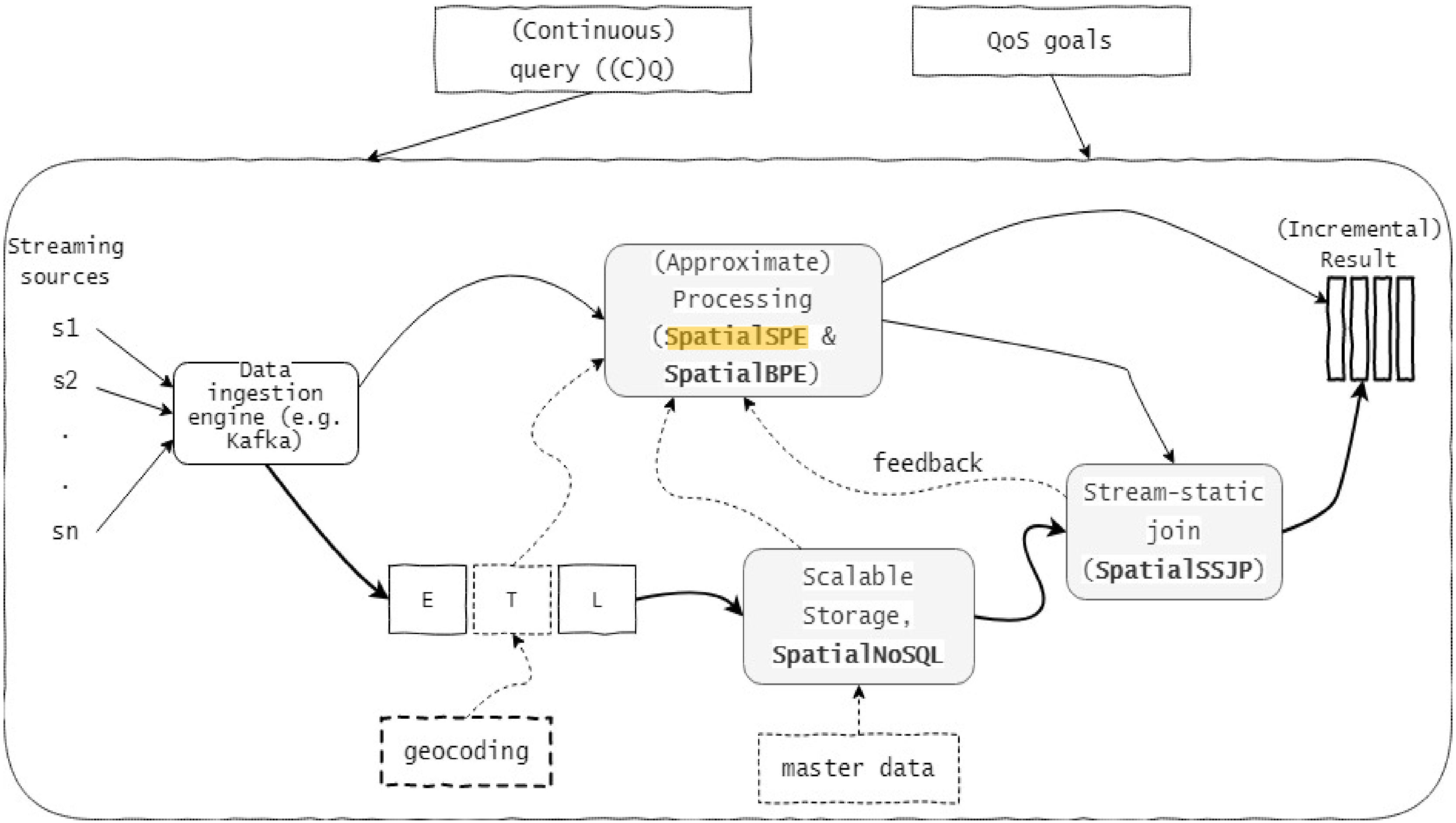
Lower better



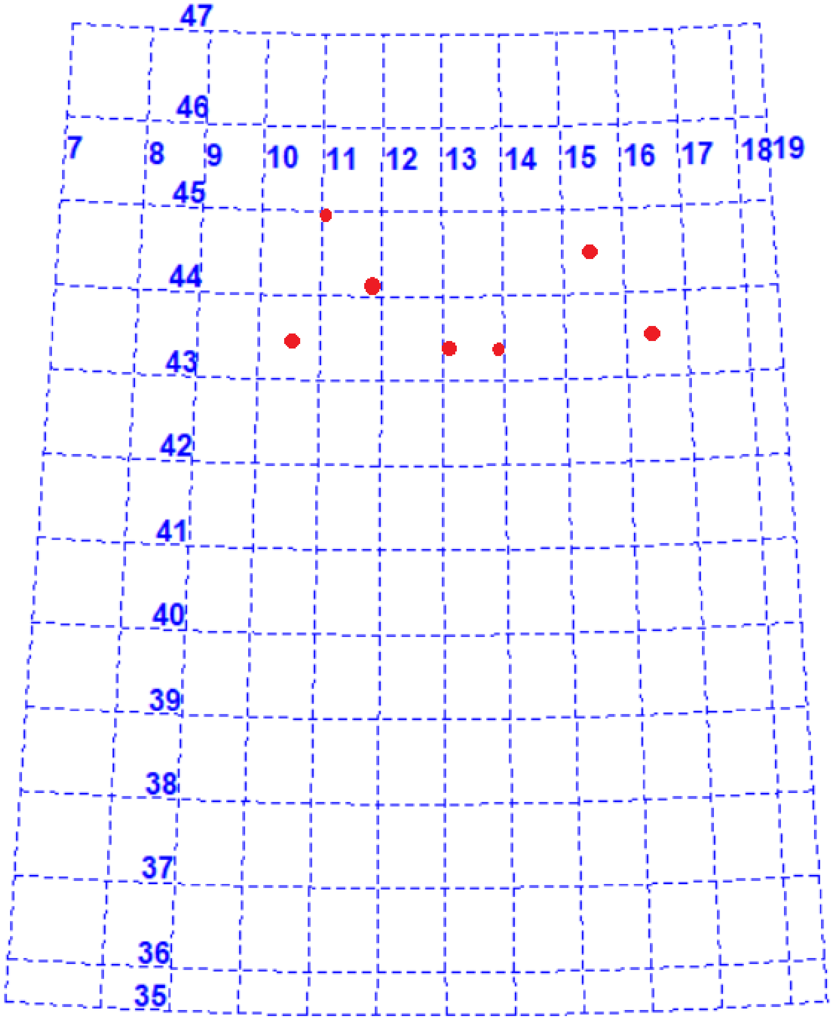
Lower better

- **NY City** taxicab trips datasets, a cohort of two months (around three million units)
- Geohash 30, our geohash-based query optimizer searches **three shards only**
- Documents and keys examined using our optimizer are **less** than those using plain MongoDB

QoS-aware interactive big spatial data processing

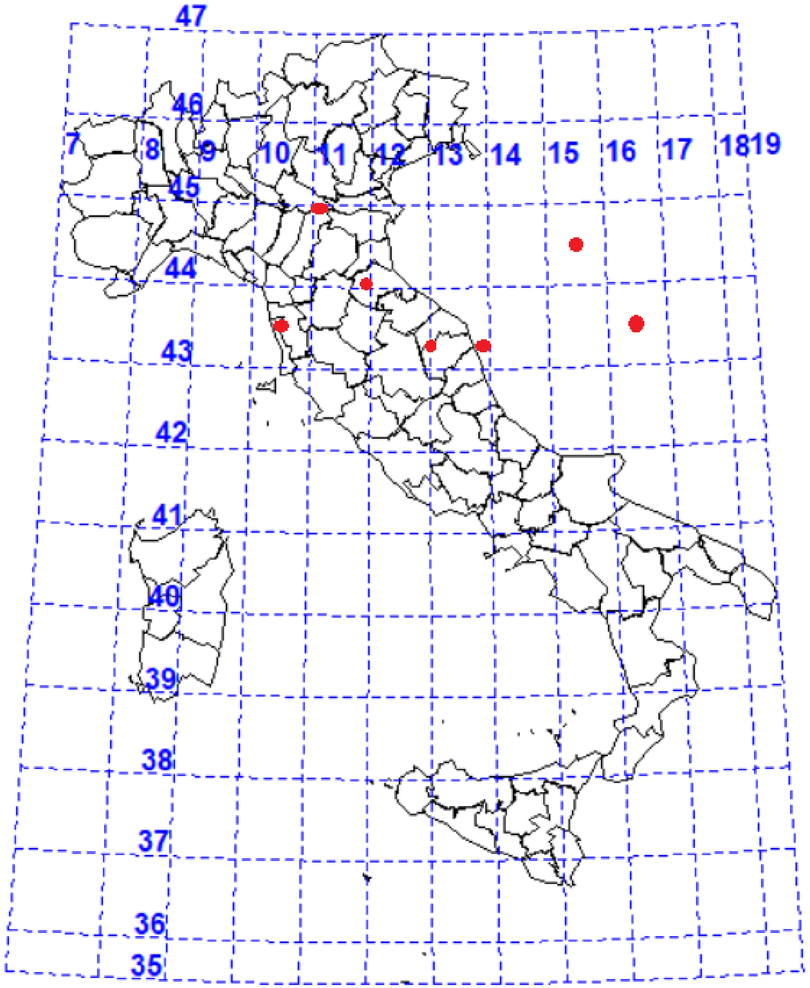


Parametrized spatial data

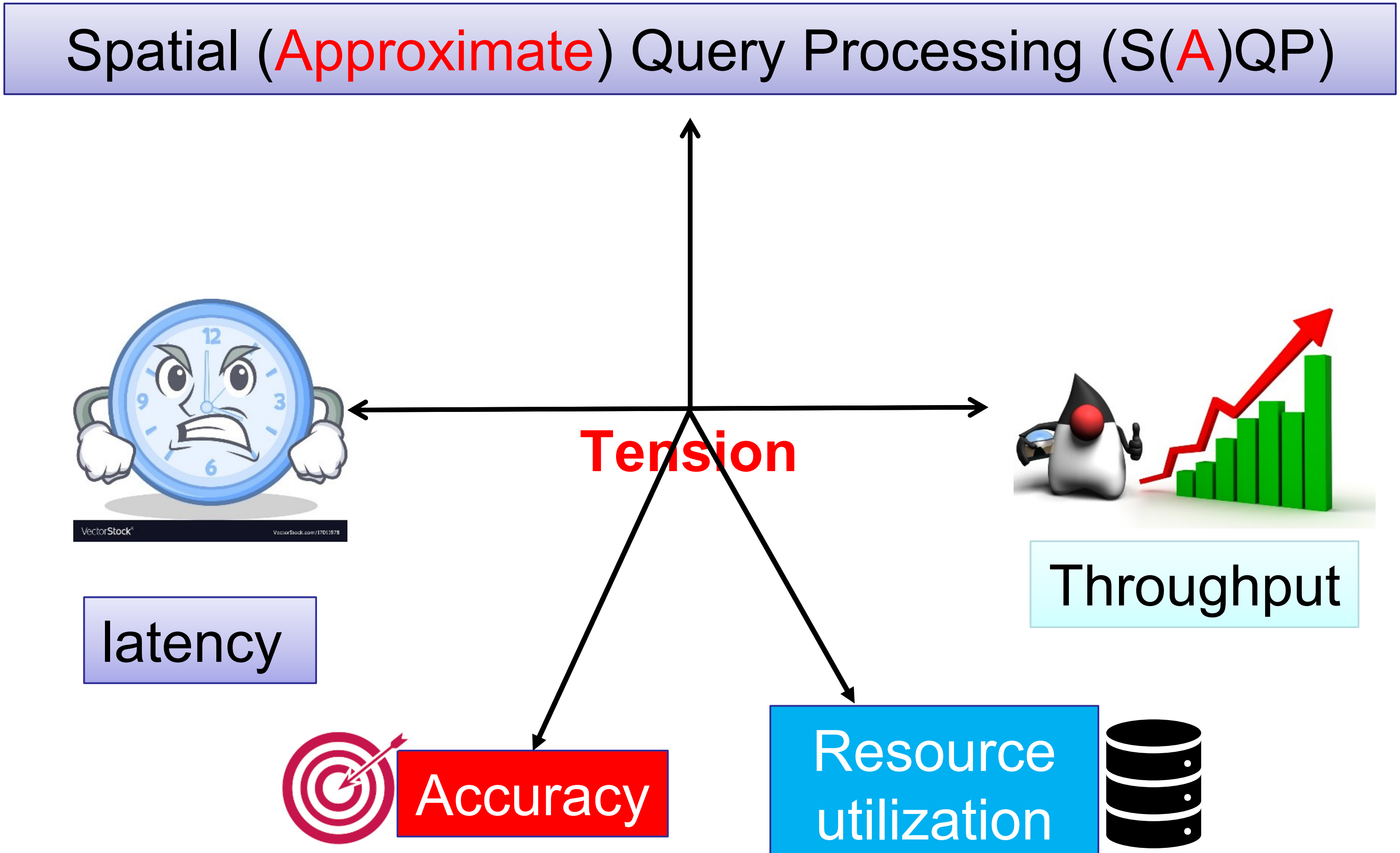


Embedding area polygons

Overlaying maps

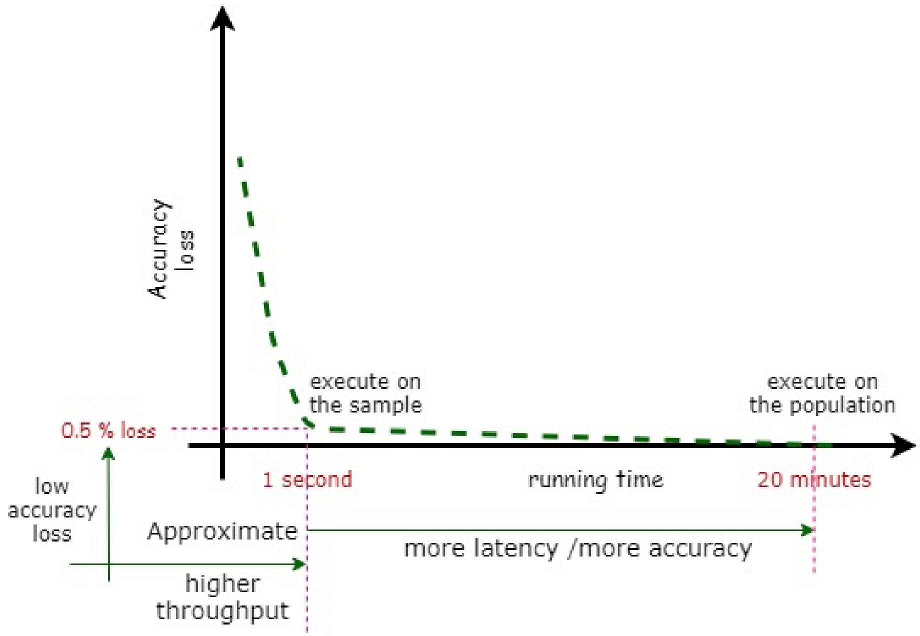


QoS Tension


Spatial (Approximate) Query Processing (S(A)QP)
Tension
VectorStock
latency
Throughput
Accuracy
Resource utilization

Spatial Approximate Query Processing (SAQP)

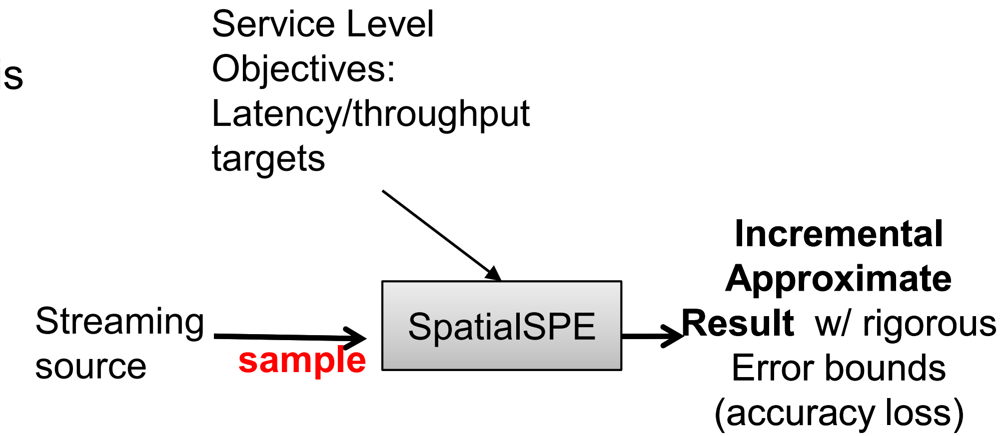
- Stream Processing Engines (SPEs) are confronted with complex challenges
  - ➢ Fast arriving streaming workloads
  - ➢ Temporal arrival rate fluctuation and skewness
- Can we do better?
  - ➢ After 1 second, we obtain a 99.5 accurate early result, which is satisfactory for decision making, which then makes the final exact result not needed



SpatialSPE

- Spatial data maintain spatial trends that affect the observed responses
  - ***spatially representative samples*** → selecting spatially well-spread out samples positively affects the accuracy of estimators (average, median, etc.).
  - A sample constitutes a scaled-down ('microcosm') version , mirroring characteristics of the population it is representing
- ***Example Continuous Query (CQ).*** "measuring the average trip distance travelled by taxis from each borough in NYC, United States"
- Sampling fractions are the same for all constituent stratum
- CQ is ***incrementalized.***

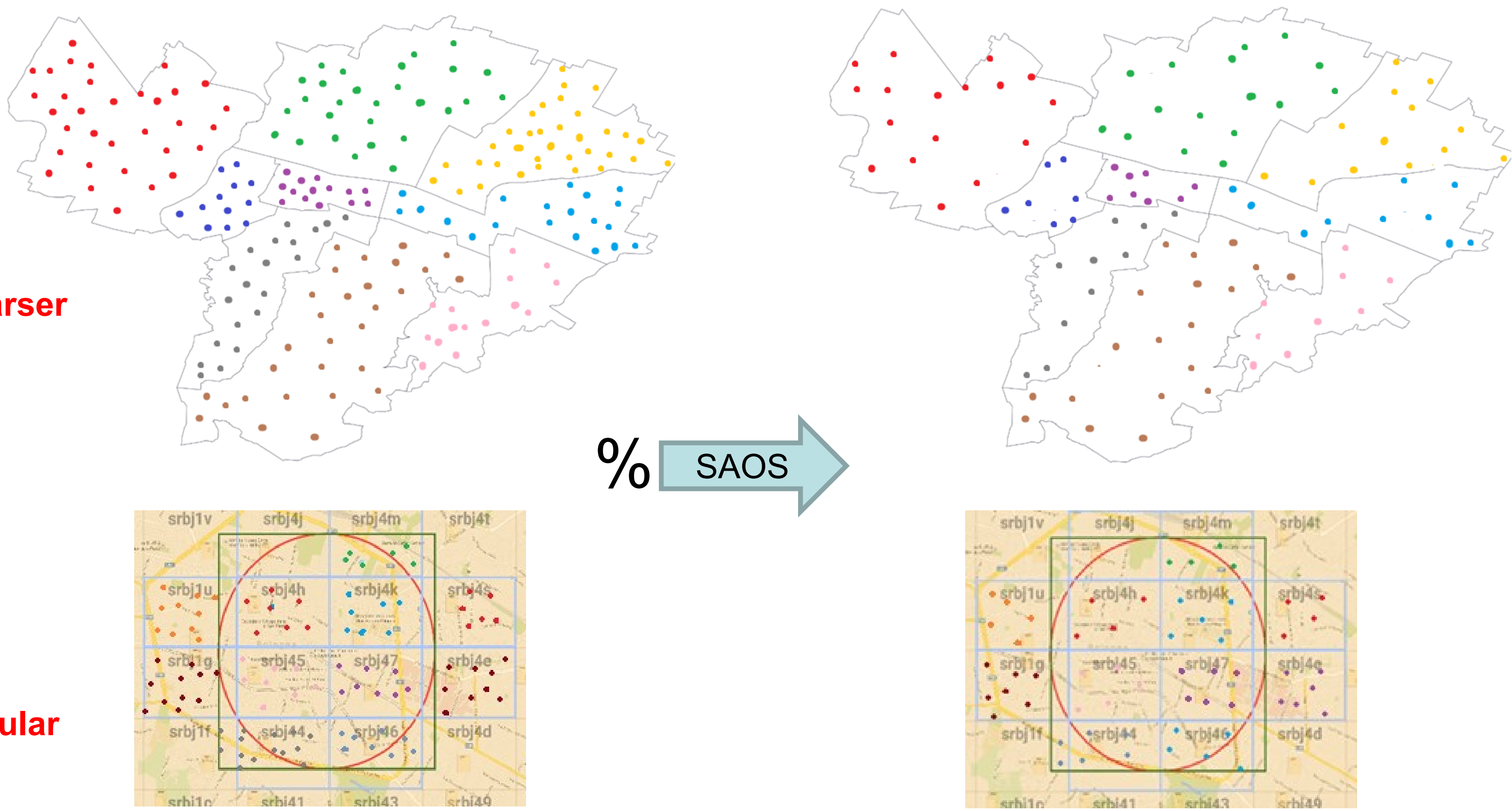
Computing over a sample instead of the whole population



SpatialSPE overview

Spatial Aware Online Sampling (SAOS): overview

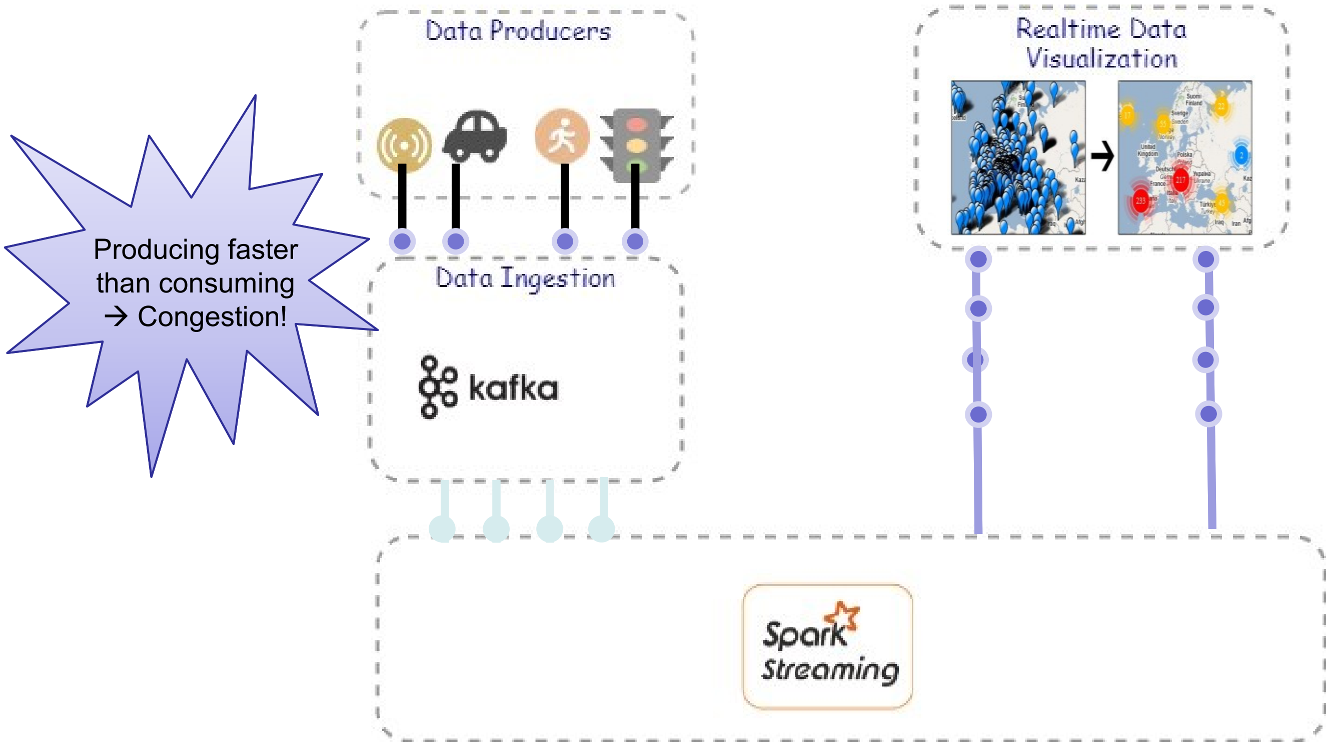
Coarser



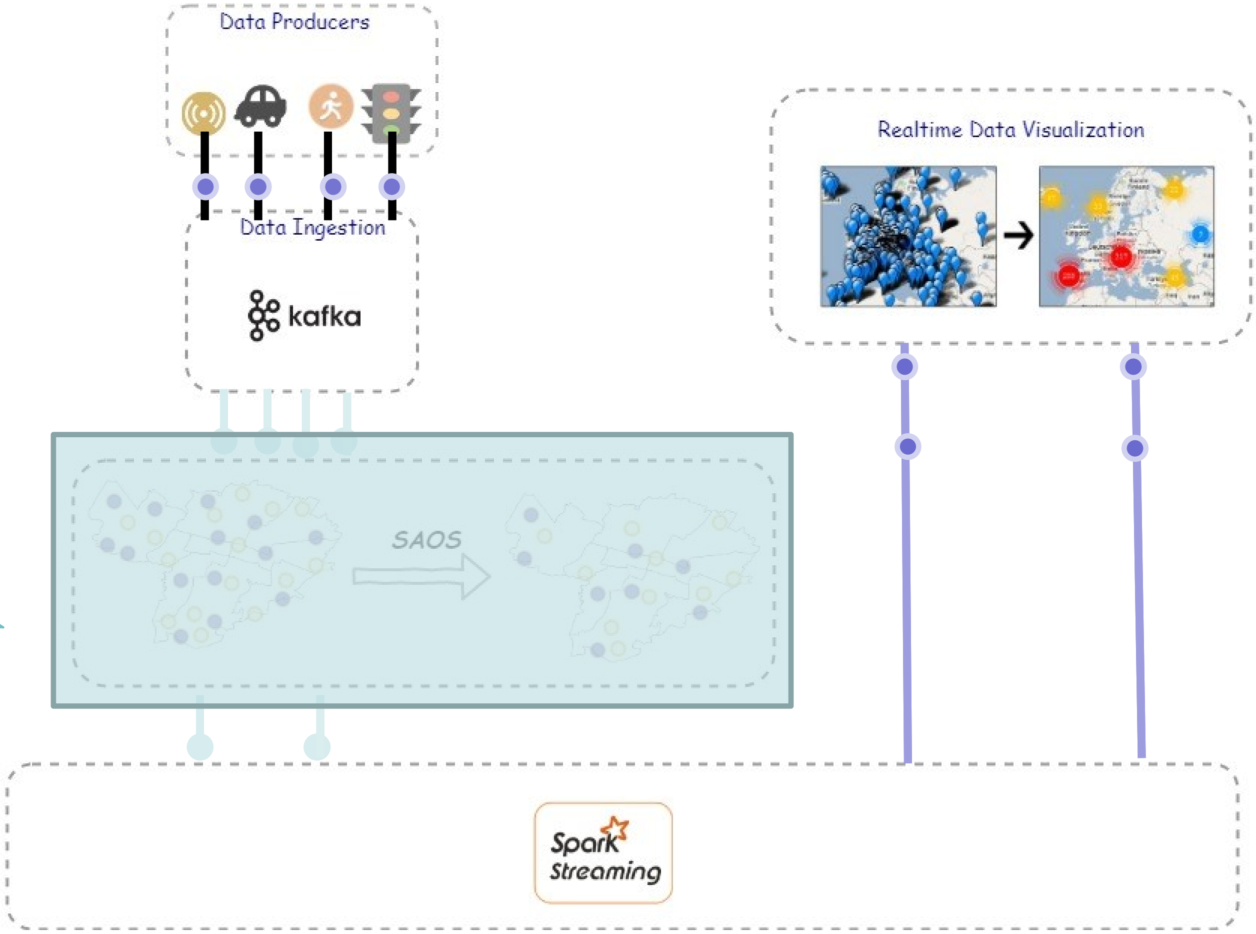
Granular

- Nearby points share the same geohash prefixes
- **SAOS** focuses on **SDL preservation**

Typical pipeline architecture w/o SAOS



The improved architecture w/ SAOS

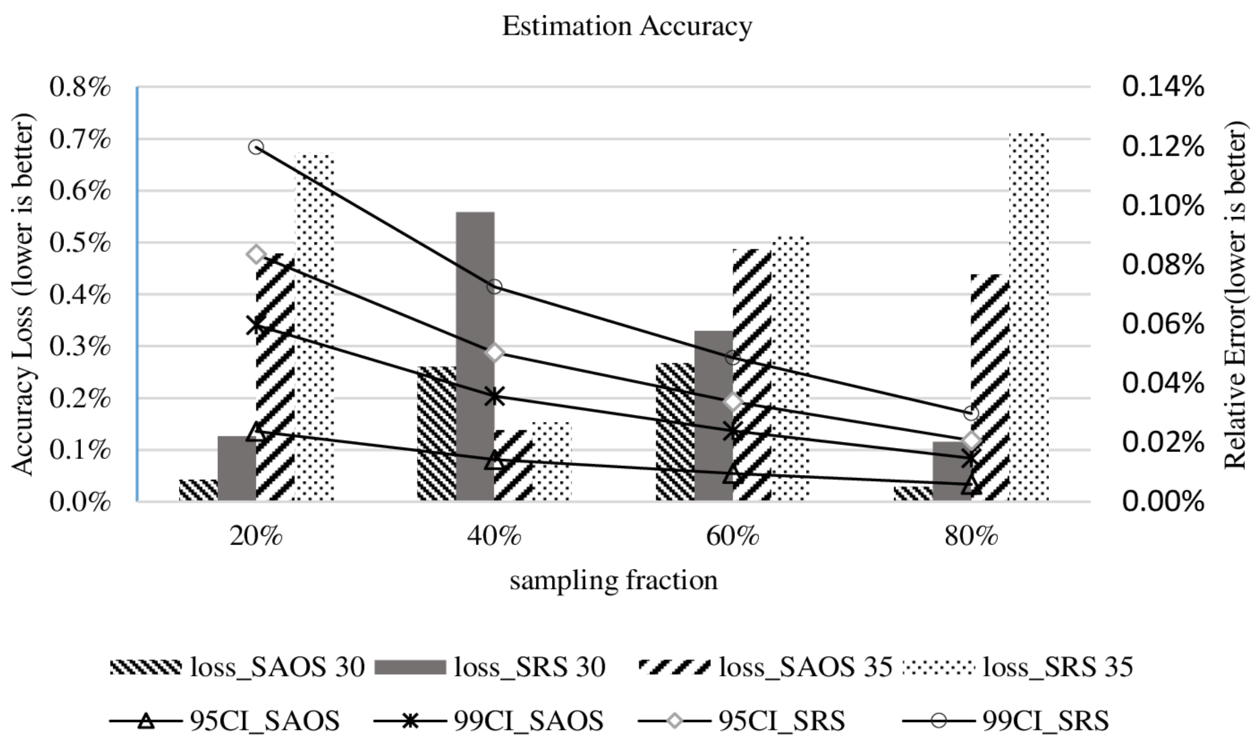


SQL-alike continuous geospatial query

Same serving speed, but serving less data!

Results: achieving accuracy targets

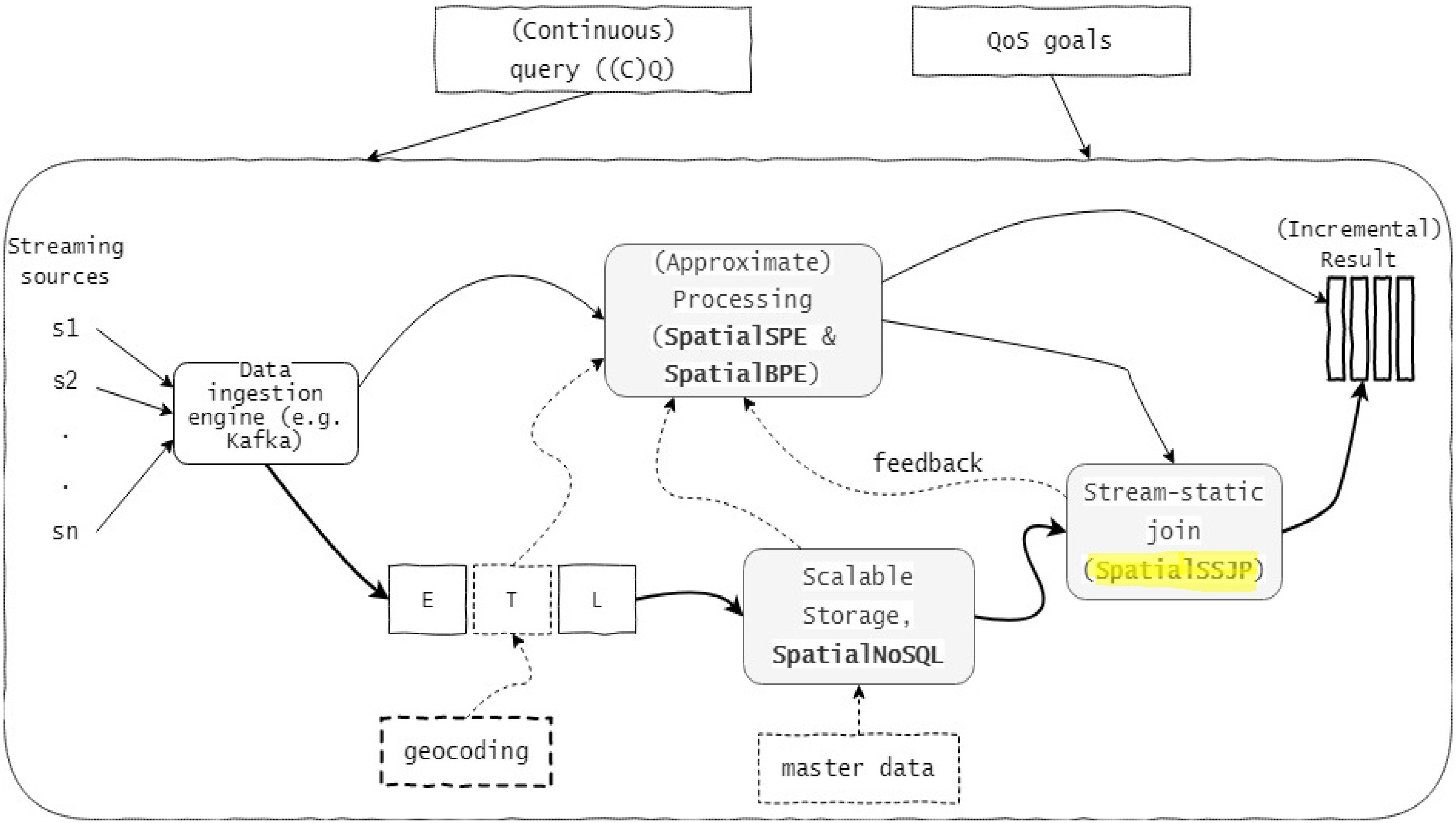
Lower better



Lower better

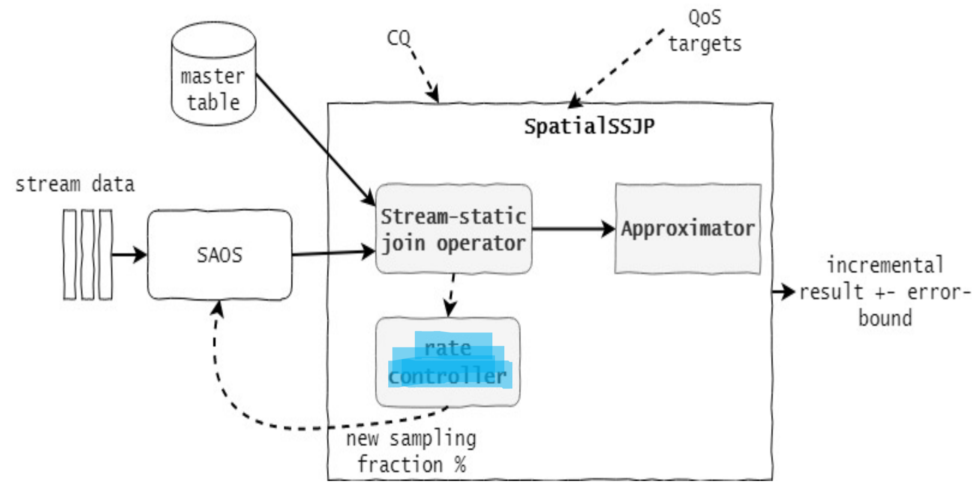
- **NY City** green taxicab trips datasets , where we select a big cohort representing six months (almost nine million tuples)
- SAOS outperforms SpSS-based SRS for all precision settings (30 and 35), for both measures, accuracy loss and relative error.
- SAOS have bigger accuracy loss for geohash precision 35 , compared to SAOS accuracy loss at geohash precision 30.

SpatialDSMS: a platform for geo big data



SpatialSSJP Overview

- Adaptive QoS- and Spatial-aware framework for processing spatial stream-static joins
  - **Continuous spatial query** requiring stream-static join and a **query budget** are served
  - **Rate controller** computes sampling fraction
  - **SAOS** samples based on the fraction
  - Stream-static **join operator** result is fed to an approximator
  - **Approximator** serves an incremental result with rigorous error-bounds
- Hybridizing a novel rate controller with SAOS from SpatialSPE
- Either latency or accuracy QoS guarantees




```
SELECT point p, polygon po, avg(tripDistance)
FROM Stream S, MasterTable M
WHERE S.key = M.key AND (p WITHIN po)
GroupBy neighborhood
Latency 120 MS
OR
e 0.03 CL 95%
```

Rate controller

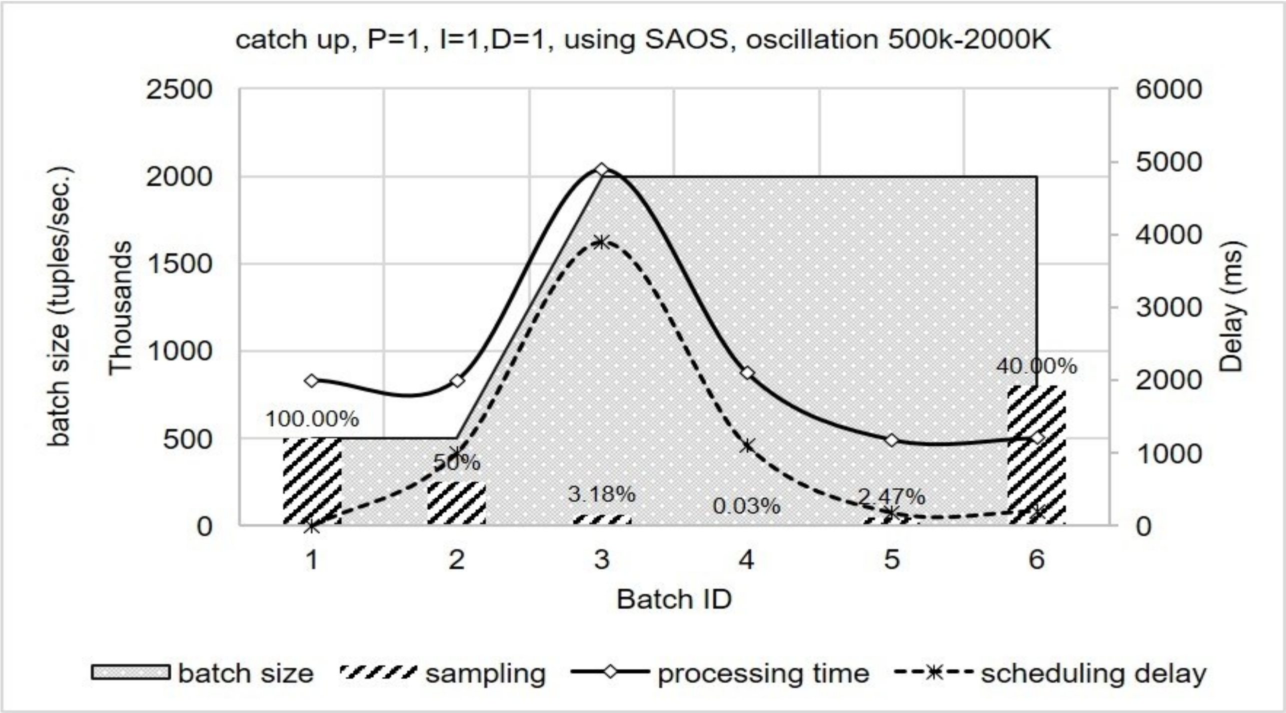
Latency-aware rate controller

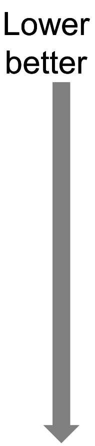
- PID controller is a control loop feedback mechanism that calculates an error value
- Three terms (PID):
  - ***Proportional****:* present error
  - ***Integral****:* historical errors
  - ***Derivative****:* future errors
- After each trigger, the new rate is calculated
  - $rate_{new} = rate_{latest} - \big((P.err) + (I.err_{hist}) + (D.err_{d})\big) = SP - PV$

Accuracy-aware rate controller

- '*margin of error'* specified as a QoS target
- For SAOS, we depend on theory of stratification, for 95% confidence level, we compute the sampling fraction:
  - $n = 3.84 * (v/{e_{des}}^{2})$

Results: achieving latency goals

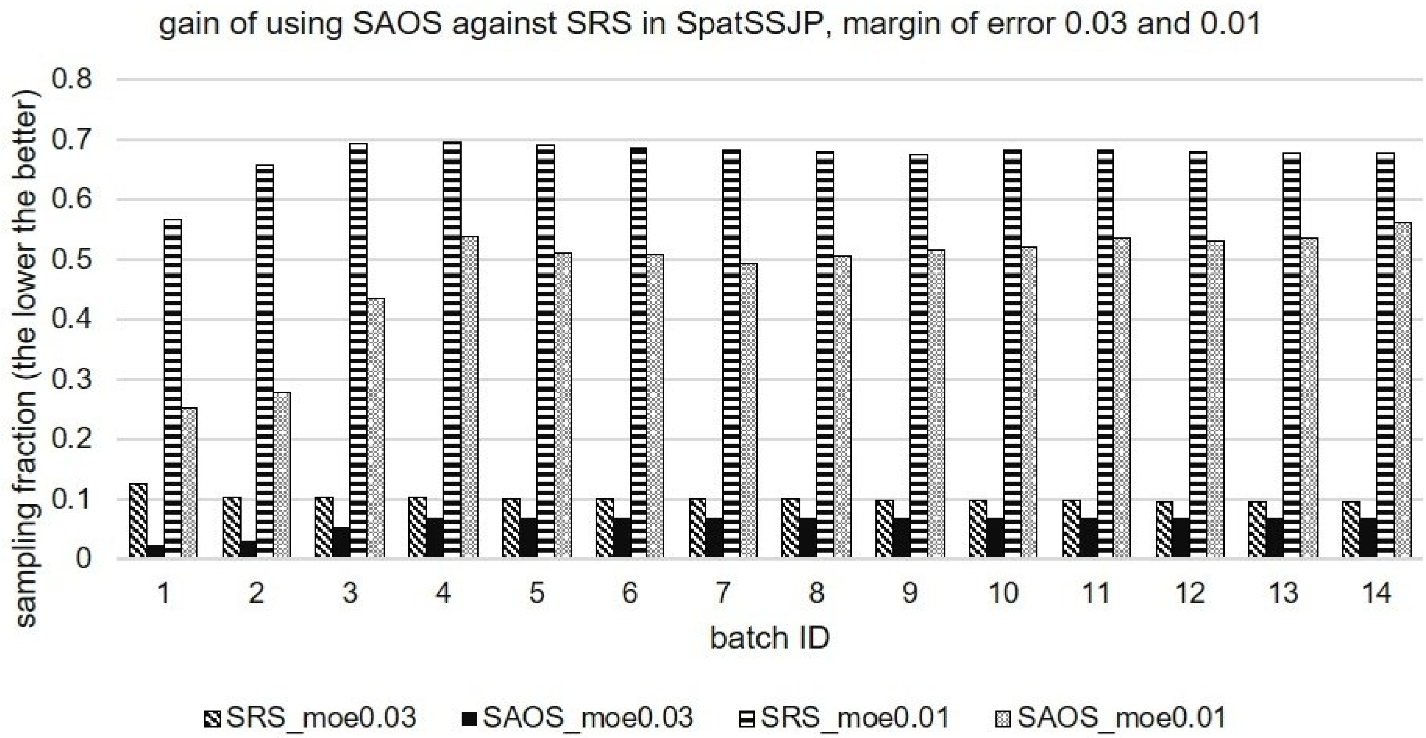




- **NY City** green taxicab trips datasets , where we select a big cohort representing six months (almost nine million tuples)
- SpatialSSJP could survive brutal spikes in the data stream arrival rates

Results: achieving accuracy goals

Lower better

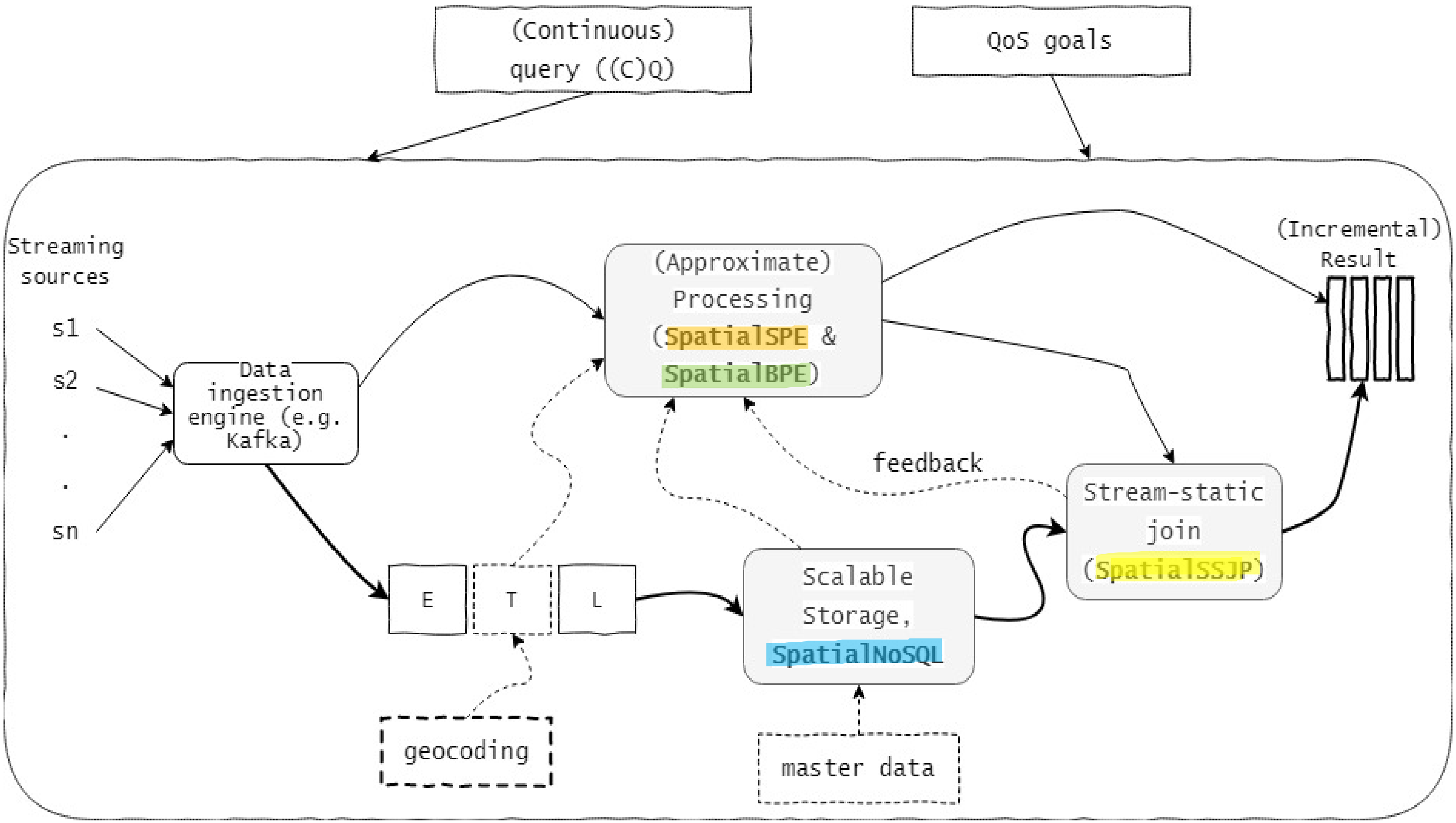


- SRS-based or SAOS achieve **accuracy** targets
- However, SAOS requires, on average, **less sampling fractions** → lower latency and higher resource utilization
- **Restrictive** cases (margin of error 0.01 imply more sampling fractions)

Outline

- ~~Motivation and thesis statement~~
- ~~Thesis contribution: a novel architecture for QoS-aware distributed processing of big spatial data~~
  - ~~Processing and storage of static data~~
    - ~~QoS-aware spatial batch processing engine~~
    - ~~QoS-aware scalable storage for spatial data~~
  - ~~Online Spatial Approximate Query Processing~~
    - ~~QoS-aware stream processing engine for spatial data~~
    - ~~Adaptive spatial stream-static join processor~~
- Conclusions and future works

SpatialDSMS: a platform for geo big data



Summary of contributions

- Avalanches of geospatial data present businesses with formidable challenges
- “one-size-fits-all” does not hold true, data-at-rest need to be **combined** with data-in-motion for deeper insights
- Current systems do not **natively** offer QoS awareness as a transparent underlying layer for processing streams of geo-referenced data
- Constituent parts should **operate synergistically** in achieving QoS goals
- We have designed a QoS Aware DSMS for geo-referenced huge amounts of streaming data loads in **highly dynamic scenarios**, SpatialDSMS:
  - ✓ QoS-aware spatial **batch processing** engine
  - ✓ QoS-aware **scalable storage** for spatial data
  - ✓ QoS-aware **stream processing** engine for spatial data
  - ✓ Adaptive spatial **stream-static join** processor
- A modular architecture that streamlines the orchestration between the constituent sub-systems

Applicability in diverse domains

- QoS -aware optimizations we have provided in this thesis are in no way exhaustive
- They constitute precursors for other domain-specific optimizations
- mixed workloads that are easily composable by mixing some of the services we provide
  - **Real-time traffic control**
    - We offer **baselines** for building a fully-functional real-time traffic control system: *flow rates*, *occupancy* and *density*
    - We support **incrementalization** for a primitive set of spatial statistics
  - **Spatial online stream clustering**
    - Composability : using the baseline primitives we support
    - Online clustering algorithms work by combining two phases:
      - **Online**: *incrementally* cluster data points based on proximity, forming ***micro-clusters***
      - **Offline**: ***macro-clustering***, forming actual clusters in batch mode using an advanced clustering algorithm

Open research areas

- **Designing** online spatial-aware data partitioning schemes
  - Taking spatial partitioning goals online imposes challenges that do not affect batch partitioning
- **Offloading** sequential jobs to **Fog** nodes
  - Sending endlessly huge amounts of geo-referenced loads to the **cloud** which could be detrimental in low-latency applications
  - Example, an online **sampler** can be pushed upstream near the Edge, achieving better latency QoS goals

Relevant publications

[1] **I. M. Al Jawarneh**, P. Bellavista, L. Foschini and R. Montanari, "Spatial-aware approximate big data stream processing," in 2019 IEEE Global Communications Conference (GLOBECOM), 2019, pp. 1-6.

[2] **I. M. Al Jawarneh**, P. Bellavista, A. Corradi, L. Foschini, R. Montanari and A. Zanotti, "In-memory spatial-aware framework for processing proximity-alike queries in big spatial data," in 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2018, pp. 1-6.

[3] **I. M. Al Jawarneh**, P. Bellavista, F. Casimiro, A. Corradi and L. Foschini, "Cost-effective strategies for provisioning NoSQL storage services in support for industry 4.0," in 2018 IEEE Symposium on Computers and Communications (ISCC), 2018, pp. 1227.

[4] **I. M. Aljawarneh**, P. Bellavista, C. R. De Rolt and L. Foschini, "Dynamic identification of participatory mobile health communities," in Cloud Infrastructures, Services, and IoT Systems for Smart Cities. Springer, 2017, pp. 208-217.

[5] **I. M. Aljawarneh**, P. Bellavista, A. Corradi, R. Montanari, L. Foschini and A. Zanotti, "Efficient spark-based framework for big geospatial data query processing and analysis," in 2017 IEEE Symposium on Computers and Communications (ISCC), 2017, pp. 851-856.

[6] **I. M. Aljawarneh**, P. Bellavista, A. Corradi, L. Foschini, R. Montanari, "Efficient QoS-Aware Spatial Join Processing for NoSQL Scalable Storage Frameworks". 2020. **Submitted**.

[7] **I. M. Aljawarneh**, P. Bellavista, A. Corradi, L. Foschini, R. Montanari, “. SpatialSSJP: QoS-Aware Adaptive Approximate Stream-Static Spatial Join Processor”. 2020. **Submitted**.

[8] **I. M. Aljawarneh**, P. Bellavista, A. Corradi, L. Foschini, R. Montanari, " Locality-Preserving Spatial Partitioning Scheme for Quality Spatial Analytics in Distributed Main Memory Frameworks".2020. **Submitted**.

[9] **I. M. Aljawarneh**, P. Bellavista, A. Corradi, L. Foschini, R. Montanari. “QoS-Aware Optimizations for Big Geospatial Data Management - A Survey”. 2020. **Submitted.**

Other publications

[10] **I. M. Al Jawarneh**, P. Bellavista, A. Corradi, L. Foschini, R. Montanari, J. Berrocal and J. M. Murillo, "A Pre-Filtering Approach for Incorporating Contextual Information Into Deep Learning Based Recommender Systems," IEEE Access, vol. 8, pp. 40485-40498, 2020.

[11] S. Bertacchi, **I. M. Al Jawarneh**, F. I. Apollonio, G. Bertacchi, M. Cancilla, L. Foschini, C. Grana, G. Martuscelli and R. Montanari, "SACHER project: A cloud platform and integrated services for cultural heritage and for restoration," in Proceedings of the 4th EAI International Conference on Smart Objects and Technologies for Social Good, 2018, pp. 283-288.

[12] **I. M. Al Jawarneh**, P. Bellavista, L. Foschini, R. Montanari, J. Berrocal and J. M. Murillo, "Toward privacy-aware healthcare data fusion systems," in International Workshop on Gerontechnology, 2018, pp. 26-37.

[13] **I. M. Al Jawarneh**, P. Bellavista, F. Bosi, L. Foschini, G. Martuscelli, R. Montanari and A. Palopoli, "Container orchestration engines: A thorough functional and performance comparison," in ICC 2019-2019 IEEE International Conference on Communications (ICC), 2019, pp. 1-6.

[14] **I. M. Al Jawarneh**, P. Bellavista, L. Foschini, G. Martuscelli, R. Montanari, A. Palopoli and F. Bosi, "Qos and performance metrics for container-based virtualization in cloud environments," in Proceedings of the 20th International Conference on Distributed Computing and Networking, 2019, pp. 178-182.

[15] P. Bellavista, J. Berrocal, A. Corradi, S. K. Das, L. Foschini, **I. M. Al Jawarneh** and A. Zanni, "How Fog Computing Can Support Latency/Reliability-Sensitive IoT Applications: An overview and a Taxonomy of State-Of-The-Art Solutions," 2019.

to the real heroes!

A Tutti Voi... Grazie!

By:

Isam Mashhour Hasan Al Jawarneh,

PhD Candidate, Cycle XXXII

Department of Computer Science and Engineering - DISI
University of Bologna, Italy
isam.aljawarneh3@unibo.it

Supervisor:
Prof. **Rebecca Montanari**

PhD program coordinator:
Prof. **Davide Sangiorgi**

Esame finale anno 2020
2nd April 2020