Efficient Generation of Approximate Region-based Geo-maps from Big Geotagged Data

Isam Mashhour Al Jawarneh[1], Luca Foschini[2] , Antonio Corradi[2]

[1] *Department of Computer Science, University of Sharjah,* Sharjah P. O. Box 27272, United Arab Emirates

[2]*Dipartimento di Informatica – Scienza e Ingegneria, University of Bologna,* Viale Risorgimento 2, 40136 Bologna, Italy

ijawarneh@sharjah.ac.ae, luca.foschini@unibo.it, antonio.corradi@unibo.it

***Abstract*— smart city applications scenarios, such as traffic monitoring, require regularly generating region-based geographical maps (geo-maps) such as choropleth, to uncover statistical patterns in the data, therefore helping municipalities to achieve better urban planning. However, with tremendous avalanches of big data arriving in fast streams, it is becoming cumbersome and inefficient to achieve the visualization task in a timely manner. Having said that, spatial approximate query processing presents itself as an indispensable and reliable solution in cases of data overloading. In this paper, we focus on presenting a novel system for generating efficiently region-based geo-maps from fast arriving big georeferenced data streams. We specifically present ApproxGeoViz. It is a system for generating approximate region-based maps from fast arriving data relying on a novel stratified-like spatial sampling method. We have built a standard-compliant prototype and tested its performance on real smart city data. Our results show that ApproxGeoViz is efficient in terms of time-based and accuracy-based QoS constraints such as running time and approximate map accuracy.**

Keywords—heat maps and choropleth, earth mover's distance, geospatial visualization, spatial data sampling, geohash, approximate query processing

I. INTRODUCTION

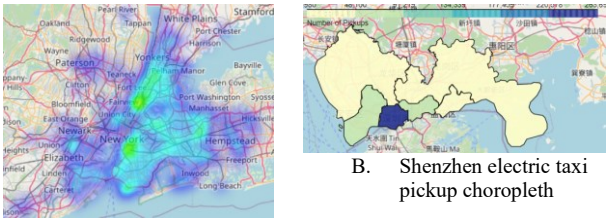
Huge amounts of georeferenced data streams are gathered hourly from actively-pulsating smart cities, being that in a form of mobility data, pollution data or any other geotagged data [1-3]. This is attributed normally to the fact that billions of GPS-enabled handheld devices are nowadays becoming ubiquitous [4]. This data is normally subjected to all kinds of spatiotemporal data science tasks, such as Exploratory Spatial Data Analytics (ESDA). An indispensable task in ESDA is the task of generating regularly geo-maps (e.g., region-based maps such as choropleth), which helps in smart city urban planning [5, 6]. For example, Fig.1.A shows a heatmap of green taxi traffic density during one month in NYC in USA, while Fig.1.B shows a choropleth map showing distributions of electric taxi pickups in the city of Shenzhen in China.

With fast-arriving fluctuating-in-nature big geo-referenced data streams, it is becoming a challenging and costly task to generate region-based maps on a regular basis (e.g., every few seconds), to a point that in brutal spikes in arrival rate, it may overload the system and easily bring it into a halt. To avoid such cases that cause the system to become out-of-service, spatial approximate query processing (SAQP) is becoming essential [6, 7]. In this paper, we show the design and implementation of a novel system for the generation of high-quality approximate region-based geo-maps. We specifically focus currently on choropleth maps that are very common in smart city and urban planning scenarios.

There are two main tasks in generating region-based geo-maps. (1) preprocessing geospatial data, where data need to be fetched and spatial queries are executed (e.g., aggregation queries such as group-by neighborhood) to bring the geospatial tuples that will be geo-visualized. (2) Geo-map visualization, that enforces a geo-map visualization effect, e.g., choropleth color encoding, over the geospatial tuples resulted from the first stage [8, 9]. The geo-visualization step is typically split into two steps: (a) translating geospatial coordinates into screen pixels, and, then (b) rendering values of features attached to those pixels to generate the corresponding image [10, 11]. It is well-corroborated in the literature that the plain vector data size has a clear correlation with the pixel data rendering cost, in such a way that bigger data size implies higher rendering cost.

The two main stages are susceptible to be a prohibitive operation in case of information overloading during high spikes in multidimensional data arrival rates. For example, geo-tagged tweets during US presidential elections. Consequently, plenty of recent works in the related state-of-art have focused on optimization methods for generating efficiently geo-maps from data streams.

Approaches for generating region-based geo-maps can be divided into two kinds, those that generate exact maps using all the data arriving, and those that are based on approximation, by sampling or taking sketches, or any other valid data size



A. NYC taxi pickups heatmap

B. Shenzhen electric taxi pickup choropleth

Fig. 1. Region-based geo-maps

reduction or compression approaches. The former is more accurate, however is more costly and could be prohibitive in cases where data arrival rate exceeds the processing capacity and the visualization component of the underlying geospatial processing system. On the other hand, the latter seeks to trade-off tiny loss in accuracy for a significant gain in running times and getting timely updates of data patterns.

In this paper, we show the design and implementation of our novel system that we term as ApproxGeoViz (for approximate geo-map visualizer), which is an efficient geo-maps visualizer that hosts an indispensable component for efficiently reducing the size of the geo-data that needs to be visualized. We specifically depend on geospatial sampling, where we apply a stratified-like spatial sampling method as a front-stage to reduce the size of the data efficiently and effectively before sending it to the geo-visualizer, which then has the direct responsibility of generating high-quality region-based geo-maps. In this paper, we specifically focus on generating choropleth maps. Our system also hosts a controller that is responsible for sensing arrival rates and decides upon percentages to drop from arriving tuples based on predefined set of accuracy-based and time-based QoS constraints. Our controller is based on Earth Movers Distance (EMD) [12].

Our geospatial sampling method is specifically based on tessellation, where we divide the geographic area (for which the region-based map is to be generated) into equal-sized rectangles using a dimensionality reduction approach that is based on Z-order curves, we specifically employ geohash encoding for this purpose as will be discussed in the next section.

The remaining sections of this paper are divided as follows. We first discuss the preliminary theory background required for subsequent discussion. We then showcase the design and prototyping of our novel system ApproxGeoViz for the approximate region-based geo-maps visualization. Thereafter, we discuss the testing performance results obtained. I what follows, we conclude the paper with remarks and recommendations for future research.

II. PRELIMENARIES

In this section, we briefly discuss the preliminaries and theory background of relevant topics that are necessary to comprehend subsequent discussion about the design and implementation of our novel system in the paper.

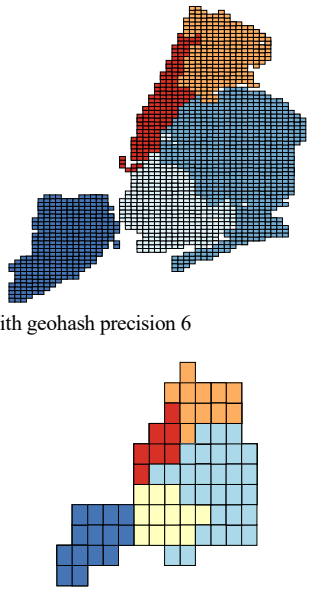
A. Geo-visualization

Geo-visualization can be loosely defined as the process of producing geo-maps from georeferenced data. It is a process that involves two main steps; (1) geospatial data processing, where geospatial queries are applied to arriving georeferenced tuples depending on the translation of the geo-visualization query. For example, for generating a choropleth map, data needs to be aggregated into clusters (pre-selected or ad-hoc), which requires applying geospatial stateful aggregation queries such as grouping by and counting tuples in each group (or finding an 'average' of a scalar value in each group, such as the 'average' speed of taxis in a taxi fleet mobility data). The outcome in this step is a dataset in a geospatial vectorized format, which then needs to be translated into a raster counterpart for the subsequent step to take place. (2) Geospatial data visualization, which is a step that requires rasterizing the vector data received from the first step and rendering it into geographical maps that can be viewed on a user screen [9]. Rasterizing vector data means finding an appropriate pixel location in the target geographical map pixel grid representation with a center that is corresponding to the geographical location of the real geospatial tuple [13].

Approaches for visualizing georeferenced data are normally grouped into three distinct categories, region-based, line-based and point-based [14]. Point-based methods plot individual points on geographical maps such as Point-of-Interest (POI) [15]. On the other hand, line-based methods generate time-series trajectory visualization of spatial data showing them moving as time ticks forward [16, 17]. From those categories, probably the region-based approaches remain the costliest in terms of the data processing step, as they rely on tessellating the geographic regions into grid cells, thereafter, grouping the data by appropriate region-based aggregations.

Choropleth maps generation is a very common typical example of visualizing georeferenced data using region-based approaches. It involves generating a map based on the predefined tessellation of the geographic region, then assigning a color density to each region based on the color coding and the density in each tile of the tessellation based on the geo-statistics or geospatial aggregations computed during the geospatial data processing phase. It is worth mentioning that geospatial aggregations for generating choropleth maps are performed on the level of regional divisions of a study area (a.k.a. administrative regions). For example, neighborhoods or districts in a metropolitan city. For example, Figure 1.B shows a choropleth map of Electric taxi pickups in the city of Shenzhen in China. Polygons (i.e., neighborhoods) that have darker blue colors have the highest density of taxi pickups.

Heatmaps is another common example of region-based geo-maps that are common in smart cities. Despite some discrepancies, all kinds of region-based geo-maps require stateful data aggregation, which is known to be computationally expensive in real data stream settings and could easily bring systems out-of-service in cases of brutal spikes in data arrival rates. This is so because georeferenced data is typically parametrized while moving it over networks to reduce the network pressure. Parametrization means representing geospatial points as longitude/latitude pairs of coordinates. This has the consequence that geospatial tuples lose their real geometrical representation in this kind of representation. Performing geospatial aggregation to generate region-based maps require brining those parametrized points into their original forms, thus specifying to which regions in real geometries they belong, which is a kind of geospatial join that is computationally costly in data stream settings [18]. Having said that, it is obvious that a dominating component in the process of generating geospatial region-based maps is geospatial data preprocessing. In that sense, if preprocessing is



A. With geohash precision 6

B. With geohash precision 5

Fig. 2 Geohash tessellation for NYC city, USA

taking much time than it should, a resort is to rely on geospatial AQP such as load shedding and geospatial sampling.

B. Geospatial Data Modeling, Reduction and Sampling

Working with huge amounts of multidimensional data streams requires two things to guarantee processing data within thresholds of QoS guarantees. Those are geospatial data representations that are succeeded by imposing access structures on those representations [19, 20]. Geospatial data representation models can be classified into two main kinds: space-driven and data-driven. The space from which data is withdrawn is normally referred to as the *embedding space*. Space-driven works by partitioning the embedding space (geographical area from which data is withdrawn), akin to order-preserving hash functions, using, for example, quadtrees and grid files. On the other hand, data-driven is based upon a partitioning of the data items themselves, using, for example, R-trees and KD-trees. For space-driven grids, those can be either regularly-sized equal grids or arbitrarily-sized grids. This data representation is typically followed by spatial data structures that assist in speeding up the access to target data according to the spatial representation and distribution of data [21]. Stated another way, to efficiently work with voluminous geospatial data, it is indispensable to first represent the data using the appropriate multidimensional model such as those that are tree-based, thereafter a spatial data structure is imposed on the model representation to provide speed access to data, such as using B+-trees.

To further enhance the modelling and access of spatial data, modern Geographic Information Systems (GISs) apply what is known as 'ordering', which is an approach to reduce multidimensional data so that it comes down to single-dimensions, an example includes Z-order curves. Ordering is imposed onto the embedding space representation (e.g., grid representation), followed by a tree-based spatial access, such as B+-trees overlays the ordering, which then provides a speed up to access the data efficiently in a timely manner.

A very important example from the family of z-order curves applications is the geohash encoding. It is basically a string representation for the cells representing the adjacent grid cell decomposition of an embedding space. Geospatial objects that have equal geohash prefixes belong normally to the same grid cell, the longer the shared prefix the closer the spatial objects in real geometries. Geohash encoding and other dimensionality reduction Z-order-based approaches, such as Googles S2 and Ubers H3, are very important tools for speeding up the processing of deluges of geospatial tuples. Geohash in this sense is employed as a quick-and-approximate filter for spatial proximity scans. Figure 2 shows geohash covering generated for NYC in USA at a geohash precision that equals 6, whereas figure 2.B shows the geohash covering for same city with geohash precision 5. Precision is the number of characters in the string representation for a geohash value. For example, 'dr5rux' is the value of one of the grid cells covering NYC in figure 2.A, where 'dr5ru' represents one of those cells covering the NYC at precision 5 as shown in figure 2.B.

Geohash is a dimensionality reduction approach that is useful for geocoding geospatial points as short strings of a length between one and twelve. Longer Geohash has a granular precision (covering smaller area).

One of the most important tools in AQP techniques is sampling. In statistics, sampling is loosely defined as the procedure of selecting a representative subset of a population for estimating an unknown population parameter value, such as an '*mean'* or '*count'*.

The sampling design is the process by which a sample of units or sites is selected. However, there is an agreement that the sample must be representative of the population from which it is withdrawn. In other terms, sample is a scaled-down version of the population capturing and mirroring appropriately the characteristics of the population it is representing.

The target of obtaining a perfectly-representative sample is unattainable. However, a sample that captures the reality in a way that helps to render the traits of the study variables to a plausible degree of accuracy and confidence is what we seek normally. One of the typical challenges that cause some sampling designs to be considered bad is what is known as 'selection biasedness', for which as sampling method overlooks few parts of the population by design [22].

There are two core sampling designs in the literature, simple random sampling (hereafter SRS for short) and stratified sampling (SS for short, in what follows). SRS assigns equal selection probability to every unit in a population, it then assigns labels to every unit and selects labels randomly, up to the point that predetermined number of unique units, which is equal to the sample size, is captured. Stratified-based sampling designs select equal of non-equal portions from each distinct group in data, for example 50% males and 50% females from

student's population in a school [22]. Stratified-like sampling designs are preferred over other counterparts for the overarching traits they offer as they are known to yield better estimations as compared to random-based counterparts [22].

With fast-arriving deluges of voluminous overwhelming big geo-referenced data streams, it is becoming less convenient to seek deterministic answers for complicated spatial queries and geo-map visualization in real-time. Add to this the facts that geospatial data is multidimensional with complex data structures and show oscillation in data arrival rates and skewness, then the problem is further inflamed. In the arena of geo-statistics and geo-visualization, Spatial AQP solutions that capture approximations with error-bounds are highly appreciated in the literature [23].

Bearing that in mind, relying on a spatial sampling design that is based on stratification for selecting samples from fast-arriving spatial data streams is efficient and known to work preferably in the literature. It is the best design that can be exploited for generating region-based approximate geo-maps from voluminous geotagged data. It is also preferable over deterministic solutions because, for most of the real-life scenarios, the process of observing all objects in a population could be impossible and intractable, for example observing migrating birds in a large geographic span.

III. ApproxGeoViz: Geospatial Visualization at Scale with QoS Guarantees

In this section, we show the design and implementation of our novel system ApproxGeoViz for the efficient generation of region-based geo-maps from big geotagged locational data, with a specific focus in this paper on choropleth maps.

Our novel system is composed of five main components as shown in the schematic context diagram of figure 3. Those components work synergistically in a workflow as a pipeline that operates in a left-right order. Those components are as follows: *geospatial data modelling and representation* module, *stratified-like geospatial sampler*, *region-based geo-map proxy generator*, *geo-map renderer* (visualizer), and *QoS controller*. The mechanism by which our system operates is the following. Two types of raw geotagged data are served to the modelling and representation module; those are the raw geotagged big data tuples (on the order of millions or even billions) and a polygon file representing the study area from which raw tuples where collected, typically on the form of either geojson file or

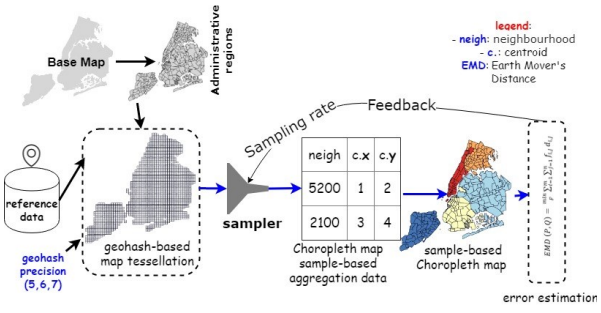


Fig. 3 ApproxGeoViz architecture

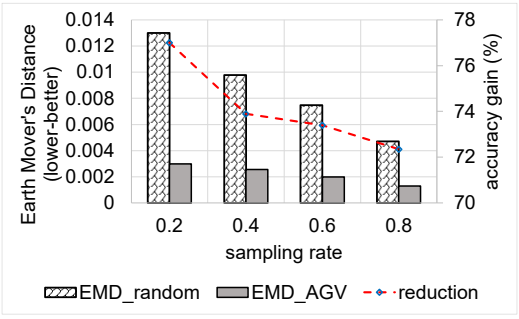


Fig. 4 EMD ApproxGeoViz (AGV in figure legend) applying stratified-like sampler against random sampler baseline, with geohash precision 6.

shapefile. Representation module is responsible for splitting the geographic space study area (e.g., city, country, etc.,) into an equally-sized grid using geohash encoding with a prespecified precision as shown in Figure 2. An attractive feature of geohash is that it preserves spatial co-locality so that geographically nearby objects will have the same geohash values in the representation. This representation is essential for making the stratified-like sampling as we treat each individual geohash value as a stratum. The raw tuples are also geohash-encoded in this case with the same precision as the one employed for study area polygons. This representation results in two intermediate geohash-tagged datasets, one representing the polygons and the other representing the raw tuples. Each polygon is covered by many geohash values. Those intermediate datasets are then served to the *stratified-like geospatial sampler*. The sampler works as follows; for each geohash value in the geohash cover, it samples data from the raw tuples that have the same geohash representation as the one from the cover. Sampling rate is served to the system as a configurable parameter and samples are taken from each geohash bracket independently with the sampling rate, given that each polygon is represented by many geohash values, this guarantees to a good extent that roughly fair counts of tuples are sampled from each polygon independently, thus resembling stratified sampling, which is known to yield more accurate results than random sampling counterparts. Sampled data is then fed to the *region-based geo-map proxy generator*. Geo-map proxy is a compact and memory-efficient representation of aggregated georeferenced vector data on the form of a

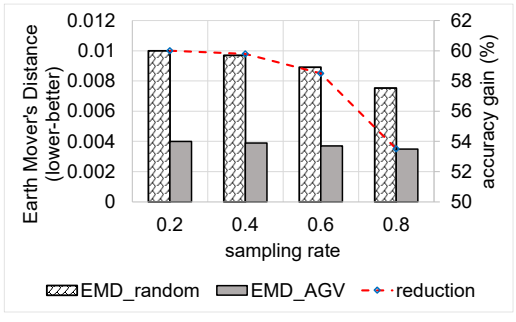


Fig. 5 EMD ApproxGeoViz (AGV in figure legend) applying stratified-like sampler against random sampler baseline, with geohash precision 5.

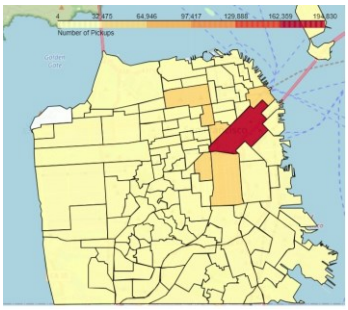


Fig. 6 Reference choropleth map SF data, no-sampling

matrix, for which each row is composed of the geo-statistic in each geohash bracket, in addition to the geohash centroid. This is a granular level, on the other hand, a coarse-grained level guarantees a similar representation by aggregating all geohash codes that cover each district (polygon) or any other coarse level of aggregation. This proxy is then passed to the region-based *geo-map renderer* (visualizer). The visualizer takes one of two directions depending on the level of granularity of the data it receives from the proxy generator. If it receives data on a coarse level on the form (polygon, geo-stat) it passes this to the plain visualizer (choropleth map visualizer for example). On the other hand, if it receives data on a fine-grained level, then it aggregates data in all geohashes covering each polygon individually, then it passes the result to the visualizer, which then proceeds by applying the plain version for choropleth map generation. The sampled vector data used in generating the approximate choropleth map is then passed to the error estimator, which applies Earth Mover's Distance (EMD) to calculate error-bounds and serves them with the map to the user.

Earth Mover's Distance (EMD) is an important information-theoretic distance metric that is normally employed for measuring similarity between two data distributions or densities. We apply (1) to calculate the EMD value.

$$EMD\ (P,Q) = \min_{F} \sum_{i=1}^{m} \sum_{j=1}^{n} flow_{i,j}\, dist_{i,j} \qquad (1)$$

Where $flow_{i,j}$ is the flow between P_i and Q_j that minimizes the total cost. $dist_{i,j}$ is the distance between P_i and Q_j (e.g., Euclidean or Manhattan distances).

IV. Experimental Evaluation

In this section, we summarize test settings, the datasets, in addition to the baseline methods and evaluation metrics.

Fig. 7 Reference choropleth map SF data, random sampling

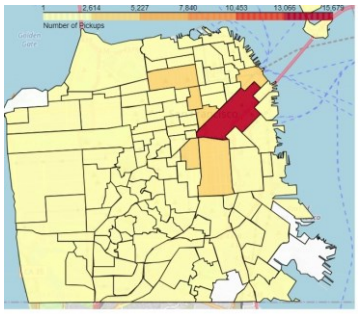


Fig. 8 Reference choropleth map SF data, stratified sampling

A. Experimental setup

Datasets. We depend on a publicly available Uber pickup dataset from the city of San Francisco in USA. It is an anonymized GPS coordinates (longitudes/latitudes) of Uber taxi trips forming around one million and 85k tuples (https://raw.githubusercontent.com/dima42/uber-gps-analysis/master/gpsdata/all.tsv, accessed on: 5 February 2023).

Deployment. We deploy our experiments on a virtual machine on Microsoft Azure with the following resources. 4 E8 v3 machines with 8 cores processors and 64 GB of RAM. We implemented the standard-compliant prototype of our system in Python utilizing geo libraries such as Geopandas.

Evaluation metrics. We use Earth Mover's Distance (EMD) distance measurement.

B. Experimental Results and Discussion

In this section we explore the usage of our ApproxGeoViz system in generating region-based maps (specifically choropleth maps) from real big geo-referenced data.

We depend on varying the geohash precision, between 5 and 6, and the sampling rate for both GeoApproxViz based on stratified-like sampler and simple random sampler (baseline) counterpart. Then we use the EMD distance measurement to test the performance of the system by applying both samplers. For geohash precision 6, we obtain results shown in Figure 4. We obtained an accuracy gain that roughly equals to 74%, ranging from 77% at sampling rate 0.2 to 72% at sampling rate 0.8. For geohash precision 5, we obtain results shown in Figure 5. We obtained accuracy gain on par with 57.9%, ranging from 60% at sampling rate 0.2 to 54% at sampling rate 0.8. This is less than the case of geohash precision 6. The explanation is the following, the smaller the geohash precision value, the bigger the coverage of each geohash and the more values it contains, this means that stratification is performed on a coarser level and more areas are involved, thus bringing the sampling design closer to the random sampling as opposed to stratified-sampling, taking into consideration that within each individual geohash cell, we sample tuples separately.

We have also tested our system performance for generating choropleth maps from sampled data, comparing stratified-like sampling to random counterpart. We provide a reference choropleth map generated from the original datasets as depicted in Figure 6. Thereafter, we use the reference data to generate two

maps, where we apply geospatial stratified sampling to proportionally sample tuples from each region independently as the new system, and the simple random sampling. We sampled 8% (in stringent stream settings where data arrival rates far exceed system processing capacity) using SRS and stratified-like sampling. Fig. 7 shows the stratified-like sampling, while Fig. 8 shows the SRS. We notice that for small sampling fractions, SRS overlooks more regions than stratified counterpart (those are colored in white in Figures). This is an indicator that a stratified-sampling in the front-stage is more adept in capturing a picture that resembles the real geometries more accurately, thus more accuracy is accrued.

V. Conclusions and Future Works

In this paper, we have shown the design and implementation of a novel system ApproxGeoViz for the efficient generation of approximate aggregate-based geo-maps from fast arriving georeferenced data streams. ApproxGeoViz employs a stratified-like sampling method at the front-stage as a quick-and-approximate filter that intelligently discards extra loads of data in cases where the system geo-visualizer is unable to keep standing with the pace of data arrival rates. Specifically, ApproxGeoViz employs a QoS controller that computes (after every few rounds, or batch intervals) the appropriate sampling rate and serves that to the sampler in the front-stage in a loop feedback mechanism approach. This guarantees sampling tuples that can be efficiently geo-visualized given the capacity of the system. Currently, the value of the similarity is arbitrarily-selected or expert-guided, we aim to fill this gap by contributing a future work where we develop a mathematically-principled algorithm to decide upon the similarity value based on data stream characteristics and statistics. ApproxGeoViz currently runs on independent servers. We aim at exploring methods to design a version that runs in distributed computing environments, probably atop frameworks such as Apache Spark.

Acknowledgment

This research was supported in part by the OpenModel project and has received partial funding from the European Union‘s Horizon 2020 research and innovation program under Grant Agreement No 953167.

References

[1] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, and R. Montanari, "Locality-Preserving Spatial Partitioning for Geo Big Data Analytics in Main Memory Frameworks," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*, 2020: IEEE, pp. 1-6.

[2] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, and R. Montanari, "Efficient QoS-Aware Spatial Join Processing for Scalable NoSQL Storage Frameworks," *IEEE Transactions on Network and Service Management,* 2020.

[3] I. M. Al Jawarneh, L. Foschini, and P. Bellavista, "Efficient Integration of Heterogeneous Mobility-Pollution Big Data for Joint Analytics at Scale with QoS Guarantees," *Future Internet,* vol. 15, no. 8, p. 263, 2023.

[4] D. C. de Oliveira, J. Liu, and E. Pacitti, *Data-Intensive Workflow Management*. Springer Nature, 2022.

[5] A. Hassan and J. Vijayaraghavan, *Geospatial Data Science Quick Start Guide: Effective techniques for performing smarter geospatial analysis using location intelligence*. Packt Publishing Ltd, 2019.

[6] I. M. A. Jawarneh, L. Foschini, and P. Bellavista, "Polygon Simplification for the Efficient Approximate Analytics of Georeferenced Big Data," *Sensors,* vol. 23, no. 19, p. 8178, 2023.

[7] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, and R. Montanari, "QoS-Aware Approximate Query Processing for Smart Cities Spatial Data Streams," *Sensors,* vol. 21, no. 12, p. 4160, 2021.

[8] J. Yu, A. Tahir, and M. Sarwat, "GeoSparkViz in action: a data system with built-in support for geospatial visualization," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, 2019: IEEE, pp. 1992-1995.

[9] J. Yu, "Src: geospatial visual analytics belongs to database systems: the babylon approach," *SIGSPATIAL Special,* vol. 9, no. 3, pp. 2-3, 2018.

[10] M. Guo, Y. Huang, Q. Guan, Z. Xie, and L. Wu, "An efficient data organization and scheduling strategy for accelerating large vector data rendering," *Transactions in GIS,* vol. 21, no. 6, pp. 1217-1236, 2017.

[11] M. Guo, Q. Guan, Z. Xie, L. Wu, X. Luo, and Y. Huang, "A spatially adaptive decomposition approach for parallel vector data visualization of polylines and polygons," *International Journal of Geographical Information Science,* vol. 29, no. 8, pp. 1419-1440, 2015.

[12] S. T. Rachev, "The Monge–Kantorovich mass transference problem and its stochastic applications," *Theory of Probability & Its Applications,* vol. 29, no. 4, pp. 647-676, 1985.

[13] M. Ma, Y. Wu, X. Ouyang, L. Chen, J. Li, and N. Jing, "HiVision: Rapid visualization of large-scale spatial vector data," *Computers & Geosciences,* vol. 147, p. 104665, 2021.

[14] Y. Zheng, W. Wu, Y. Chen, H. Qu, and L. M. Ni, "Visual analytics in urban computing: An overview," *IEEE Transactions on Big Data,* vol. 2, no. 3, pp. 276-296, 2016.

[15] F. Miranda *et al.*, "Urban pulse: Capturing the rhythm of cities," *IEEE transactions on visualization and computer graphics,* vol. 23, no. 1, pp. 791-800, 2016.

[16] S. Al-Dohuki *et al.*, "Semantictraj: A new approach to interacting with massive taxi trajectories," *IEEE transactions on visualization and computer graphics,* vol. 23, no. 1, pp. 11-20, 2016.

[17] X. Huang, Y. Zhao, C. Ma, J. Yang, X. Ye, and C. Zhang, "TrajGraph: A graph-based visual analytics approach to studying urban network centralities using taxi trajectory data," *IEEE transactions on visualization and computer graphics,* vol. 22, no. 1, pp. 160-169, 2015.

[18] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, and R. Montanari, "Efficiently Integrating Mobility and Environment Data for Climate Change Analytics," in *2021 IEEE 26th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2021: IEEE, pp. 1-5.

[19] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, and R. Montanari, "Spatially Representative Online Big Data Sampling for Smart Cities," in *2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2020: IEEE, pp. 1-6.

[20] 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: IEEE, pp. 1-6.

[21] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, and R. Montanari, "Big Spatial Data Management for the Internet of Things: A Survey," *Journal of Network and Systems Management,* vol. 28, no. 4, pp. 990-1035, 2020.

[22] S. L. Lohr, *Sampling: design and analysis*. Nelson Education, 2009

[23] N. Stoehr *et al.*, "Heatflip: Temporal-Spatial Sampling for Progressive Heat Maps on Social Media Data," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018: IEEE, pp. 3723-373