

# Advancements of Randomized Data Structures for Geospatial Data

Paul Walther

*Supervised by Martin Werner*

*Professorship for Big Geospatial Data Management, Technical University of Munich, Munich, Germany*

## Abstract

Increasing amount of geospatial data acquired by various sensory improvements and increased governmental and societal interest in creating this data, results in rising issues in data storage and processing for this data type. Therefore this work describes the setting for a probabilistic data structure based on Random Projections and Bloom Filters to enable a more efficient processing of geospatial data. The paper states four research questions, which, apart from general considerations on probabilistic data structures, elaborate on the advantages of such data structures in their use together with new hardware like field programmable gate arrays (FPGA) or quantum computers. The main contribution is to structure the problem space and define solution approaches for future research in this field.

## Keywords

geospatial data, geo-data management, randomized data structures, bloom filter, geographic information systems

## 1. Introduction

In recent years, the amount of data generated has increased with the ability of technical devices to capture and store this information as well as governmental and societal willingness to create this data persistently. In the geospatial domain, which combines location information with attribute and sometimes temporal information [1], this became especially apparent as mobile technical devices, such as phones, and earth observation devices, like satellites and drones, create increasing data streams. This imposes significant challenges on the data processing solutions as pure data generation does not imply application-oriented usage. On the one hand, global analyses are challenging due to the huge storage footprints of the generated data [2], and on the other hand, edge computing solutions, like direct processing on drones or satellites, are challenged with processing only the data they generate, not speaking about setting this data in context to external data streams.

Especially on a global scale, the evaluation of single small areas is often inefficient: The represented data is frequently sparse, and information on neighboring regions is often not independent [2]. Furthermore, acquired spatial information often has no exact location but only an approximate one [3]. Still, current procedures result in huge datasets, which are hard to query if, e.g., statistics based on single sparse elements need to be calculated. Improvements can be achieved either through

compression and simplification of the data or improved query processing during data analysis to access the data more efficiently in big data environments. However, for geospatial data and data queries, only a few compression algorithms, dimension reduction techniques, or database access patterns were developed, which will be described in the section 2.


To proceed based on this, my doctoral research will focus on improving the fundamentals of randomized data structures and query processing to make them more applicable to geospatial data and larger databases. This includes theoretical considerations based on information theory, implementations to achieve efficiently running algorithms, and the consideration of hardware implications. The final goal is to propose a new application-proofed randomized data structure for geospatial data.


## 2. Related Work


Geospatial data, in general, can appear in different formats such as land surveys, GPS information, photography, remote sensing images, digitized maps, point clouds, and more [3]. Today's standards for storing geometric data include point cloud representations, simplicial complexes, topological approaches, and CityGML [4], which are mostly stored in object-oriented [5], object-relational [6, 7, 8] or graph-based databases [9]. The most used representation is a form of simplicial complexes; the rasterization of the data [10]. This means that the to-be-described area or volume is quantized in distinct spatial pixels/voxels, each associated with one or more attribute values. One way to improve the efficiency of the storage is to reduce the data footprint. For normal databases, this can be done in two ways: Reduce the numerosity of the

*Published in the Proceedings of the Workshops of the EDBT/ICDT 2024 Joint Conference (March 25-28, 2024), Paestum, Italy*

 paul.walther@tum.de (P. Walther)

 0000-0002-5101-5793 (P. Walther)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

samples or reduce the dimensionality [11]. We can fit a probabilistic model to the data to learn a mapping to a lower dimensional subspace.

The probably easiest way to do so is a random projection, which poses a way of projecting data to a lower dimensional subspace by picking a random subset of the properties [12]. Another probabilistic data structure is a Bloom Filter (BF) as shown in Figure 1 [13]. BFs,

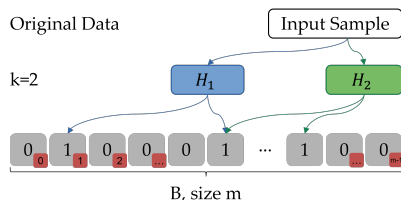


Figure 1: Visualization of Bloom Filter Approach

which were initially developed in the 1970s, represent a probabilistic randomized data structure to represent sets, trading data footprint, and computational effort against error probability [2]. For construction, a bit array  $B$  of size  $m$  is set to one at the indexes denoted by the result of a set of  $k$  linear evenly distributed hash functions  $H_k$  mapping the input sample to integer numbers  $< m$ . For membership queries, a possible sample is vice versa, again mapped with the same  $H_k$  to a set of indexes. If the BF  $B$  is zero at one of these indices, the tested sample is not part of the set; otherwise, it is with a certain probability in the set. Special about this structure is that it does not allow False Negatives in such membership queries [14]. Therefore, if False Positives are acceptable or can be mitigated by other means the use of a BF-based data structure is a system with a controllable error rate [2]. The downside of this data structure is that, in its initial form, it does not allow the deletion of objects from a computed set. Therefore, various modifications of this approach were proposed in the past, e.g., counting BFs, which try to mitigate this issue but often introduce False Negatives [15].

Using BFs to represent geospatial data was rarely reported before: E.g., [2] proposes a way to compress sparse binary global datasets with BF-like data structures in main memory and keep pixel-wise accessibility. Their proposition approves existing solutions in specific cases, like the pixel-wise random access of temporally ordered but spatially labeled information. For random projections, there is no previous work reported for geospatial data to the best of our knowledge. Contrarily using BFs for efficient set queries is widely used in various fields [14] but not explicitly reported for spatial data to the author's knowledge.

### 3. Research Questions

The main goal of the project is the development of a new data structure for geospatial data leveraging the advantages of randomized data structures. This leads to the following research questions (RQs):

- RQ<sub>1</sub> How can a randomized data structure be modified for the **efficient storage and retrieval** of geospatial data?
- RQ<sub>2</sub> How is a **spatial movement** of the stored data efficiently represented in terms of database modifications?
- RQ<sub>3</sub> What are the properties of a geospatial database to enable continuous **scaling of the database storage footprint**?
- RQ<sub>4</sub> How do the newly proposed geospatial data representations leverage the **computation on new hardware** like FPGAs and quantum computers?

### 4. Methods

Based on the research questions in section 3 an improvement of spatial storage and retrieval may be achieved by further developing and adopting existing randomized data structures for this data environment. For first consideration, the work focuses on data structures based on Random Projections and BFs as described in section 2. In the following, the research questions are expanded with a planned methodology to answer them. This is the basis for finally proposing a new geospatial data structure in the future.

#### 4.1. Efficient Storage and Retrieval

Efficient storage and retrieval of geospatial data include element-wise access, deletion, and modification (later referred to as basis operations). Thereby, it has to be differentiated between the exact storage and retrieval of a single data element through proper indexing and mapping and the efficient loading of a probabilistic representation of a data element.

The aforementioned basic operations are possible for randomly projected footprints of the original data. On the contrary, a standard BF supports the addition of single elements to the data set, but the removal or modification of an element can only be conducted by a recalculation of the whole filter. Proposals of Counting BFs [16] try to mitigate this issue but are less efficient. Part of this work is, therefore, to adapt a BF architecture by either providing more efficient structures for geospatial data based on the counting BF architecture or adding an external modification data structure that keeps track of modifications and triggers recalculations of the filter or parts of the filter based on a to be defined schedule or rule.

Another nonbasic operation that will be considered in the work is the merging or linking of several datasets through probabilistic data structures. This is especially important for the computation in distributed systems where different devices take over different operations, and their results must be merged. Finding the right merge keys poses a challenging task. Conversely, simple bitwise OR operations can merge the BF representation of data sets. However, there are issues with this approach: The optimal BF construction is dependent on the number of samples to be represented. As the total number is not expected to be known during the independent construction of the two datasets, the combined filter may increase FP rates. Therefore, it is necessary to investigate how to modify the BF to perform merges efficiently, especially on representations of geospatial data.

## 4.2. Spatial Movement

Moving geo data spatially, which is stored in probabilistic databases, is a topic that has not been discussed so far (to our knowledge). At the same time, it is an often performed operation in the geospatial domain. For data structures based on random projections, a movement of the input data in the spatial domain has to result in a similar movement of the projection. With rasterized data  $X \in \mathbb{R}^a$  a lower dimensional space  $S \in \mathbb{R}^b$  with  $a \gg b$ , a random projection  $R$  and a movement in the spatial domain  $M$  this mean it should hold

$$x = R(X) \in S \rightarrow R(M)x = R(MX) \quad (1)$$

To still achieve efficient random projections it needs to be investigated how this changes the set of randomly selected points to project.

For BFs, it has to be discussed how a spatial movement of the data can be expressed in a modification of the BF, such that a simple movement of the data does not require a recalculation of the whole BF representation. Therefore, an investigation of how certain spatial patterns can be encoded by efficient use of the hash functions or an extension of the standard BF to additional hierarchies is to be investigated. The main challenge here is the dimension difference between spatial locations (2D or more) and standard 1D BFs. The ideas here involve either the mapping of the input data in 1D, e.g., through space-filling curves, or encoding the spatial domain by mapping spatial data points only to subsets of the final BF. The second enables a spatial movement by applying simple bit-shift operations on the stored data as shown in Figure 2. Alternatively, a second dimension can be added to the BF, which encodes the location separately or locality-sensitive hashing techniques may be used.

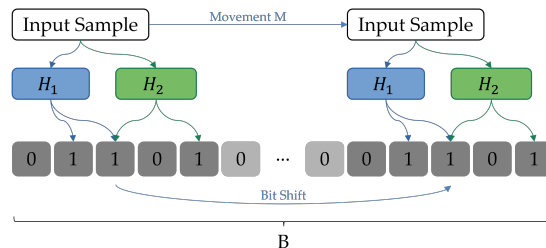


Figure 2: Visualization of a bit shift in the Bloom Filter B to represent a movement  $M$  in the spatial domain

## 4.3. Scaling of the Data Base Footprint

Both considered randomized data structures, Random Projections, and BFs, have the useful property of a relatively easy resizing. For Random Projections, a fraction of the randomly picked elements hold still a fraction of the initial information. Taking only a subset of the randomly projected points results in a representation that keeps the initial information. Questionable in this case is how to choose the subset most efficiently.

For BFs, it is defined by construction that taking one-half of the filter holds a valid representation to which additional data can still be added. However, scaling with a factor  $\neq 2$  is not easily possible [2]. This is especially important as today's hardware often also scales with 2, e.g., 8 GB  $\rightarrow$  16 GB RAM. Therefore, reducing the size of a, e.g., 64 GB large data set by a Factor of  $\frac{1}{2^n}$  is not always efficient as it leaves too much or no space for system processes. Therefore, scaling by a rational factor should be introduced. This may be conducted by the superposition of non-uniformly distributed hash functions.

For very small representations of the data, this fractioning is not as easy for BFs: As there can be computed an optimal amount of hash functions based on the amount of to-be-stored samples and the requested filter sized for very small filters, often a non-natural amount of hash functions would be optimal. Therefore, the author proposes to use a rational number of Hash Functions by applying the functions with a probability only representing the rational fraction of the number.

## 4.4. Computation on New Hardware

For the computation on future hardware especially small representations of the to be considered data are needed. Such, FPGAs have to deal with limited storage space. This means embedding the original data fitting to the hardware requirements is needed. The same is true for quantum computers, which are still constricted to a limited amount of Qubits. By using randomly projected representations of the original complex geo-data, it is, therefore,

still possible to compute in this hardware-restricted environment. Furthermore, BFs can map the original data to a bit array. This is especially helpful for the computation with quantum computers as it enables a simple and gate-efficient data embedding by only using non-complex basis encoding.

## 5. Conclusion and Future Work

Based on the above considerations and propositions a new method to store probabilistic representations of geospatial data sets is to be developed. This will enable the efficient usage of big geospatial data in new application domains and on new hardware, such as the efficient route planning on remote vehicles and the efficient provision of geospatial data for quantum computers. To actually enable a broad usage of these newly proposed representations, they finally need to be integrated into a framework to use with existing GIS software. Results obtained during the development of the new data structure may also propose a new possibility to improve singular computationally expensive collection, simulation, processing, or presentation tasks of geospatial data.

## Acknowledgments

This work is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 507196470.

## References

- [1] K. Stock, H. Guesgen, Geospatial reasoning with open data, in: R. Layton, P. A. Watters (Eds.), *Algorithms for automating open source intelligence (OS-INT)*, Computer science reviews and trends, Synpress, Rockland, 2015, pp. 171–204. doi:10.1016/B978-0-12-802916-9.00010-5.
- [2] M. Werner, Globimapsai: An ai-enhanced probabilistic data structure for global raster datasets, *ACM Transactions on Spatial Algorithms and Systems* 7 (2021) 1–24. doi:10.1145/3453184.
- [3] J. Ni, C. V. Ravishankar, B. Bhanu, Probabilistic spatial database operations, in: *Advances in Spatial and Temporal Databases*, volume 2750, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 140–158. doi:10.1007/978-3-540-45072-6\_9.
- [4] M. Breunig, P. E. Bradley, M. Jahn, P. Kuper, N. Mazroob, N. Rösch, M. Al-Doori, E. Stefanakis, M. Jadidi, Geospatial data management research: Progress and future directions, *ISPRS International Journal of Geo-Information* 9 (2020) 95. doi:10.3390/ijgi9020095.
- [5] O. Balovnev, M. Breunig, A. B. Cremers, Geotoolkit: Opening access to object-oriented geodata stores, in: M. F. Goodchild (Ed.), *Interoperating geographic information systems*, Springer Science+Media Business, New York, 1999, pp. 235–247. doi:10.1007/978-1-4615-5189-8\_20.
- [6] Z. Yao, C. Nagel, F. Kunde, G. Hudra, P. Willkomm, A. Donaubaue, T. Adolphi, T. H. Kolbe, 3dcitydb - a 3d geodatabase solution for the management, analysis, and visualization of semantic 3d city models based on citygml, *Open Geospatial Data, Software and Standards* 3 (2018). doi:10.1186/s40965-018-0046-7.
- [7] M. Breunig, S. Zlatanova, 3d geo-database research: Retrospective and future directions, *Computers & Geosciences* 37 (2011) 791–803. doi:10.1016/j.cageo.2010.04.016.
- [8] H. H. Le, P. Gabriel, J. Gietzel, H. Schaeben, An object-relational spatio-temporal geoscience data model, *Computers & Geosciences* 57 (2013) 104–115. doi:10.1016/j.cageo.2013.04.014.
- [9] R. H. Güting, Graphdb: Modeling and querying graphs in databases, in: *VLDB*, volume 94, 1994, pp. 12–15.
- [10] T. Pingel, The raster data model, *Geographic Information Science & Technology Body of Knowledge 2018* (2018). doi:10.22224/gistbok/2018.3.11.
- [11] S. Kadam, D. S. Thakore, Analyzing high dimensional data mining in time series database with reduced dimensionality, *International J. of Engg. Research & Indu. Appls. (IJERIA)* 4 (2011) 421–432.
- [12] E. Bingham, H. Mannila, Random projection in dimensionality reduction, in: F. Provost, R. Srikant (Eds.), *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 245–250. doi:10.1145/502512.502546.
- [13] B. H. Bloom, Space/time trade-offs in hash coding with allowable errors, *Communications of the ACM* 13 (1970) 422–426. doi:10.1145/362686.362692.
- [14] A. Broder, M. Mitzenmacher, Network applications of bloom filters: A survey, *Internet Mathematics* 1 (2004) 485–509. doi:10.1080/15427951.2004.10129096.
- [15] A. Abdennebi, K. Kaya, A bloom filter survey: Variants for different domain applications, 2021. doi:10.48550/arXiv.2106.12189.
- [16] F. Bonomi, M. Mitzenmacher, R. Panigrahy, S. Singh, G. Varghese, An improved construction for counting bloom filters, in: Y. Azar, T. Erlebach (Eds.), *Algorithms u2013 ESA 2006*, volume 4168 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 684–695. doi:10.1007/11841036\_61.