The Era of Big Spatial Data

Ahmed Eldawy
Computer Science and Engineering
University of California, Riverside
eldawy@cs.ucr.edu

Mohamed F. Mokbel
Computer Science and Engineering
University of Minnesota
mokbel@cs.umn.edu

ABSTRACT

In this tutorial, we present the recent work in the database community for handling Big Spatial Data. This topic became very hot due to the recent explosion in the amount of spatial data generated by smart phones, satellites and medical devices, among others. This tutorial goes beyond the use of existing systems as-is (e.g., Hadoop, Spark or Impala), and digs deep into the core components of big systems (e.g., indexing and query processing) to describe how they are designed to handle big spatial data. During this 90-minute tutorial, we review the state-of-the-art work in the area of Big Spatial Data while classifying the existing research efforts according to the *implementation approach*, *underlying architecture*, and *system components*. In addition, we provide case studies of full-fledged systems and applications that handle Big Spatial Data which allows the audience to better comprehend the whole tutorial.

1. INTRODUCTION

There has been a recent explosion in the amounts of spatial data produced by several devices such as smart phones, satellites, space telescopes, medical devices, among others. This variety of such spatial data makes it widely used across important applications such as simulating the brain in the Blue Brain Project [37], identifying cancer clusters [43], tracking infectious disease [7], drug addiction [48], simulating climate changes [25], and event detection and analysis [45]. While there are several open-source distributed systems that are designed to handle Big Data in general, e.g., Hadoop, Hive [49], HBase, Spark [59], and Impala [30], they all fall short in supporting spatial data efficiently. As a result, there are great research efforts in either extending the ideas of these systems or building new systems to efficiently support Big Spatial Data.

In this tutorial, we will comprehensively go through all ongoing efforts for supporting Big Spatial Data which the authors have recently covered in a comprehensive survey [23]. We will also point out to the research challenges and opportunities in supporting big spatial data. The tutorial will act as an invitation to the database community to join arms to fill up the emerging needs of spatial big data applications. The tutorial is divided into six parts. The first part will motivate for the need to support big spatial data. The second part will discuss various architectural approaches to building big spatial data systems. The third, fourth, and fifth parts will discuss ongoing efforts and challenges in indexing, querying, and visualizing big spatial data, respectively. The tutorial is concluded by the sixth part that goes through various cases studies of systems and applications for big spatial data.

Proceedings of the VLDB Endowment, Vol. 10, No. 12

2. TUTORIAL OUTLINE

Figure 1 gives our **90-minutes** tutorial outline, which is composed of six parts, as detailed in the rest of this section. Table 1 gives a comprehensive comparison of the partial list of the papers that we will be covering during the tutorial.

Part I: Big Spatial Data: Why?

As given in Figure 1, this first part takes 10 minutes where we motivate for the need to support Big Spatial Data. We start by giving a historical background and perspective on the explosion of Big Spatial data. We then explain how existing Big Data distributed systems (e.g., Hadoop, Hive [49], HBase, Spark [59], and Impala [30]), support spatial data, and why such support is limited. To justify this lack of spatial support, we show the performance of these systems when used as-is to process spatial data as reported in various research efforts in the literature [1, 22, 28, 53, 61, 62, 64]. As a further motivation, we highlight some simple and efficient improvements that can be applied to better support spatial data.

Part II: Building a Big Spatial Data System

The second part of the tutorial takes another 10 minutes, as given in Figure 1. This part mainly covers the second, third, and fourth columns of Table 1.

We first start by categorizing all existing work into three categories based on the underlying approach for building a Big Spatial Data system, namely, *on-top*, *from-scratch* and *built-in*, while explaining each of them by examples. In the *on-top* approach, the spatial functionality is implemented through user-defined functions (UDFs), which is simple to implement but inefficient as the system core is still unaware of spatial functionality [14, 27, 28, 51, 53, 61, 62, 64]. The *from-scratch* approach builds a new system from scratch to support distributed spatial processing, which is very efficient but too complex to build and maintain [5, 15, 52, 63]. The *built-in* approach tries to balance both simplicity and efficiency by injecting spatial data awareness inside an existing distributed system [1, 22, 26, 32, 38, 54].

We will then classify existing work based on the underlying architectures as parallel databases [15, 32], array databases [44, 46, 47, 50], Key-value stores [26, 38], MPI [41, 63], MapReduce [1,2,14,19,20,22,24,54,62], and Resilient DD (RDD) [29,57]. Then, give a quick overview of high level languages for Big Spatial

- **Part I: Big Spatial Data: Why? (10 minutes)**
  - A historical perspective on the explosion of Big Spatial Data
  - The lack of systems to support Big Spatial Data
- **Part II: Building a Big Spatial Data System (10 minutes)**
  - Approaches for building a Big Spatial Data System
  - Spatial High Level Language
  - System Components
- **Part III: Indexing Big Spatial Data (20 minutes)**
  - Spatial HDFS
  - Indexing layout
  - Index Types
- **Part IV: Querying Big Spatial Data (15 minutes)**
  - Basic queries
  - Spatial Join
  - Computational Geometry
- **Part V: Visualizing Big Spatial Data (15 minutes)**
  - Single-level Images
  - Multi-level Images
- **Part VI: Case Studies (20 minutes)**
  - System cases studies
  - Application case studies

Figure 1: Tutorial Outline (90 minutes)

Data [1, 15, 21, 24, 32, 47, 52]. Although there is no much research challenges involved with this component, it is still very important as it defines how users, specially non-technical ones, are going to interact with the system. We highlight two aspects that make a proposed language more appealing to users. First, it might extend an existing language for distributed systems which makes it easier to adopt and learn by existing users [1, 21, 24]. Second, users will quickly understand and learn a language if it follows industry standards [21, 24], such as Open Geospatial Consortium (OGC) [39], which is already adopted in popular systems, e.g., PostGIS, Oracle Spatial, and ArcGIS.

This part is concluded by a quick overview of the system components, namely, *indexing*, *query processing*, and *visualization*, which will be detailed in the next three pats of the tutorial.

Part III: Indexing Big Spatial Data

As given in Figure 1, the third part lasts for 20 minutes and describes big spatial indexes (the fifth column of Table 1). We start by describing how spatial indexing is done in Hadoop Distributed File System (HDFS), since it is the standard distributed file system used in most big data systems including Hadoop, Pig [40], Hive [49], HBase, Accumulo, Spark [59], and Impala [30].

We then show that existing spatial indexes in HDFS follow a general two-level design of one *global index* that partitions data across nodes and multiple *local indexes* that organize records in each node [63]. To affirm its generality, we show how it is used to construct many index types including uniform grid [1, 22], R-tree [14, 22], R+-tree [22], Quad tree [35], PMR Quad tree [54], K-d tree [38, 47], Geohash [26], and other GiST-based indexes [33]. In addition, depending on the capabilities of the underlying system in which the index is added, we show that some of them support only static indexes [1, 22, 35, 54] in raw HDFS files, while others provide dynamic indexes [26,38] in key-value stores. After that, we show that the huge overhead of random access in HDFS [31] pushes system designers to build primary or clustered indexes [1, 22, 26, 38, 54] where the actual records are stored in the same order as the index, while there is only a little effort to implement secondary or non-clustered indexes [31, 33, 54]. This highlights an important open research problem that researchers can tackle.

Part IV: Querying Big Spatial Data

In this part, as given in Figure 1, we will spend 15 minutes in describing the different approaches in implementing different spatial queries using the spatial indexes described earlier (the sixth column in Table 1). We start by describing existing efforts for supporting *basic queries* for Big Spatial Data. Such queries include range query [1, 22, 35, 38, 54], and the family of nearest neighbor queries including k-nearest neighbor (KNN) [2, 22, 54, 61], reverse nearest neighbor (RNN) [2], and all nearest neighbor (ANN) [53]. While describing these queries, we emphasize the value of spatial indexes by showing orders of magnitude performance improvement when spatial indexes are used for the same query type.

Then, we describe existing efforts for supporting *spatial join* queries which includes self join [1], binary join [22, 62], multi-way join [27], and kNN join [34, 60]. Finally, we describe existing efforts for more advanced *computational geometry* operations for Big Spatial data such as polygon union, skyline, convex hull, farthest/closest pairs [19], and Voronoi diagram construction [2].

Part V: Visualizing Big Spatial Data

As outlined in Figure 1, we allocate 15 minutes of the tutorial to this part to talk about visualization of Big Spatial Data (the last column in Table 1). Visualizing Big Spatial Data has gained an increased importance with spatial data as opposed to non-spatial data, where it significantly helps users to explore very large spatial datasets and spot patterns that are otherwise very hard to find. While non-spatial data is usually visualized as a chart using aggregation or sampling [8, 55], spatial data requires more sophisticated visualization techniques. We categorize existing visualization techniques for Big Spatial Data into *single level* and *multi-level* visualization algorithms. Single level visualization [16, 17, 44, 51] generates a single image with a limited resolution where the quality of the image is limited by its resolution. On the other hand, multi-level images [9, 16, 17] are generated at many zoom levels where users can zoom in to see more details about the dataset, which makes it particularly suitable for big spatial data.

Part VI: Case Studies

In this final part, we spend 20 minutes to conclude the tutorial by giving a comprehensive survey of full-fledged systems and applications for Big Spatial Data, along with describing how each system supports spatial language, indexing, querying, and visualization. This part allows the audience to grasp the whole tutorial by presenting complete systems including Hadoop-GIS [1], SpatialHadoop [22], $\mathcal{MD}$-HBase [38], ESRI Tools for Hadoop [54], BRACE [52], PRADASE [35], SciDB [47], Parallel Secondo [32], GeoTrellis on Spark [29], and AsterixDB [5]. In addition, we review a number of applications that are built using these systems to support Big Spatial Data including SHAHED [17], EarthDB [44], TAREEG [4], TAGHREED [36], and AscotDB [50]. These applications show to the audience how end-user applications can utilize the systems described above to handle Big Spatial Data.

3. TARGET AUDIENCE

This tutorial targets researchers, developers, and practitioners, who are interested in processing Big Data in general, and Big Spatial Data in particular. The tutorial does not require any particular background or knowledge about spatial data. Yet, it requires basic

Table 1: Existing work in the area of big spatial data

	Approach	Architecture	Language	Indexes	Queries	Visualization
Paradise [15, 42]	From-scratch	Parallel DB	SQL	Grid	RQ, SJ, Raster	Single level
Parallel Secondo [32]	Built-in	Parallel DB	SQL-Like	Local only	RQ, SJ	-
Sphinx [18]	Built-in	Parallel DB	SQL	R-tree, Quad tree	RQ, SJ	-
SciDB [9, 44, 47]	From-scratch	Array DB	AQL, AFL	Kd tree	RQ, KNN	Single/Multi
RasDaMan [10–13]	From-scratch	Array DB	RasQL	-	Raster	Single level
$\mathcal{MD}$-HBase [38]	Built-in	KV store	-	Quad Tree, Kd tree	RQ, KNN	-
GeoMesa [26]	Built-in	KV store	CQL	Geohash	RQ	Via GeoServer
EMINC [63]	From-scratch	MPI	-	Kd tree, R-tree	RQ, K-means, DBSCAN	-
SJMR [31, 53, 61, 62]	On-top	MapReduce	-	R-tree	RQ, KNN, SJ, ANN	-
K-Means [64]	On-top	MapReduce	-	-	K-means	-
MR-DBSCAN [28]	On-top	MapReduce	-	-	DBSCAN	-
Voronoi Diagram [2]	On-top	MapReduce	-	-	VD, NN Queries	-
3D Visualization [51]	On-top	MapReduce	-	-	-	Single level
KNN Join [34, 60]	On-top	MapReduce	-	-	KNN Join	-
Multiway SJ [27]	On-top	MapReduce	-	-	Multiway SJ	-
BRACE [52]	From-scratch	MapReduce	BRASIL	Grid	SJ	-
Hadoop GIS [1]	Built-in	MapReduce	QL^{SP}	Grid	RQ, KNN, SJ	-
SpatialHadoop [16, 19, 22]	Built-in	MapReduce	Pigeon	R tree/Quad tree	RQ, KNN, SJ, CG	Single/Multi
ScalaGiST [33]	Built-in	MapReduce	-	GiST	RQ, KNN	-
Esri Tools [54]	Built-in	MapReduce	HiveQL	PMR Quad Tree	RQ, KNN	-
ISP-MC [57]	On-top	RDD	Scala-based	On-the-fly	SJ	-
GeoTrellis [29]	On-top	RDD	Scala-based	-	Map Algebra	-
GeoSpark [58]	Built-in	RDD	Scala-based	R-tree, Quad-tree	RQ, KNN, SJ	-
Simba [56]	Built-in	RDD	SQL	R-tree	RQ, KNN, SJ, KNN-Join	-
Asterix-DB [3, 5, 6]	Built-in	Hyracks	AQL	R-tree local index	RQ	-

database knowledge, which is assumed to be there for VLDB audience including junior database students. Such knowledge includes the meaning of indexing and querying terms.

4. BIOGRAPHY

Ahmed Eldawy (Ph.D., University of Minnesota) is an assistant professor at University of California, Riverside. His research interests lie in the broad area of database systems with a focus on big data management and spatial data processing. Ahmed is the inventor and main architect of the open-source SpatialHadoop project, which has been used by several companies, organizations, and research institutes around the world, and has been downloaded more than 80,000 times within one year of its release. Ahmed has many collaborators in industrial and academic research labs including IBM Watson, Microsoft Research in Redmond (MSR), and Qatar Computing Research Institute (QCRI). Ahmed serves as the Proceedings Chair of SIGMOD 2018.

Mohamed F. Mokbel (Ph.D., Purdue University) is an associate professor at University of Minnesota. His current research interests focus on database systems, GIS, and big spatial data. His research work has been recognized by the highly prestigious VLDB 10-years best paper award in 2016, by four other best paper awards, and by the NSF CAREER award 2010. Mohamed is the elected Chair of ACM SIGPATIAL for a three-years term: 2014-2017. Mohamed is the PC Co-Chair for SIGMOD 2018, was general co-chair of SSTD 2011, and program co-chair of several conferences. He is the Editor-in-Chief for Distributed and Parallel Databases Journal, on the editorial board of ACM Books, ACM Transactions on Database Systems (TODS), and ACM Transactions on Spatial Algorithms and Systems (TSAS). Mohamed has held various visiting positions at Microsoft Research, USA and Hong Kong Polytechnic University. For more information, please visit: www.cs.umn.edu/ mokbel

5. ACKNOWLEDGEMENT

This work is supported in part by the National Science Foundation under Grants IIS-1525953, CNS-1512877, IIS-0952977, and IIS-1218168.

6. REFERENCES

[1] A. Aji et al. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce. *PVLDB*, 6(11), 2013.

[2] A. Akdogan et al. Voronoi-based Geospatial Query Processing with MapReduce. In *CLOUDCOM*, pages 9–16, Nov. 2010.

[3] A. A. Alamoudi et al. External Data Access And Indexing In AsterixDB. In *CIKM*, Oct. 2015.

[4] L. Alarabi et al. TAREEG: A MapReduce-Based System for Extracting Spatial Data from OpenStreetMap. In *SIGSPATIAL*, 2014.

[5] S. Alsubaiee et al. AsterixDB: A Scalable, Open Source BDMS. *PVLDB*, 7(14), 2014.

[6] S. Alsubaiee et al. Storage Management in AsterixDB. *PVLDB*, 7(10), 2014.

[7] A. Auchincloss et al. A Review of Spatial Methods in Epidemiology: 2000-2010. *Annual Review of Public Health*, 33, 2012.

[8] L. Battle et al. Dynamic Reduction of Query Result Sets for Interactive Visualization. In *BigData*, 2013.

[9] L. Battle et al. Dynamic Prefetching of Data Tiles for Interactive Visualization. In *SIGMOD*, June 2016.

[10] P. Baumann, A. Dehmel, P. Furtado, R. Ritsch, and N. Widmann. Spatio-Temporal Retrieval with RasDaMan. In *VLDB*, Sept. 1999.

[11] P. Baumann et al. Geo/Environmental and Medical Data Management in the RasDaMan System. In *VLDB*, Aug. 1997.

[12] P. Baumann et al. The RasDaMan Approach to Multidimensional Database Management. In *SAC*, 1997.

[13] P. Baumann et al. The multidimensional database system rasdaman. In *SIGMOD*, June 1998.

[14] A. Cary, Z. Sun, V. Hristidis, and N. Rishe. Experiences on Processing Spatial Data with MapReduce. In *SSDBM*, 2009.

[15] D. J. DeWitt, N. Kabra, J. Luo, J. M. Patel, and J. Yu. Client-Server Paradise. In *VLDB*, 1994.

[16] A. Eldawy et al. HadoopViz: A MapReduce Framework for Extensible Visualization of Big Spatial Data. In *ICDE*, 2015.

[17] A. Eldawy et al. Shahed: A MapReduce-based System for Querying and Visualizing Spatio-temporal Satellite Data. In *ICDE*, 2015.

[18] A. Eldawy et al. Sphinx: Distributed Execution of Interactive SQL Queries on Big Spatial Data. In *SIGSPATIAL*, pages 78:1–78:4, 2015.

[19] A. Eldawy, Y. Li, M. F. Mokbel, and R. Janardan. CG_Hadoop: Computational Geometry in MapReduce. In *SIGSPATIAL*, 2013.

[20] A. Eldawy and M. F. Mokbel. A demonstration of spatialhadoop: An efficient mapreduce framework for spatial data. *PVLDB*, 6(12), 2013.

[21] A. Eldawy and M. F. Mokbel. Pigeon: A Spatial MapReduce Language. In *ICDE*, 2014.

[22] A. Eldawy and M. F. Mokbel. SpatialHadoop: A MapReduce Framework for Spatial Data. In *ICDE*, Apr. 2015.

[23] A. Eldawy and M. F. Mokbel. The Era of Big Spatial Data: A Survey. *Foundations and Trends in Databases*, 6(3-4):163–273, 2016.

[24] ESRI Tools for Hadoop, 2015.

[25] J. Faghmous and V. Kumar. *Spatio-Temporal Data Mining for Climate Data: Advances, Challenges, and Opportunities*. Advances in Data Mining, Springer, 2013.

[26] A. Fox et al. Spatio-temporal Indexing in Non-relational Distributed Databases. In *BIGDATA*, 2013.

[27] H. Gupta et al. Processing Multi-way Spatial Joins on Map-reduce. In *EDBT*, 2013.

[28] Y. He et al. MR-DBSCAN: A Scalable MapReduce-based DBSCAN Algorithm for Heavily Skewed Data. *Frontiers of CS*, 2014.

[29] A. Kini and R. Emanuele. Geotrellis: Adding Geospatial Capabilities to Spark, 2014.

[30] M. Kornacker et al. Impala: A Modern, Open-Source SQL Engine for Hadoop. In *CIDR*, 2015.

[31] H. Liao, J. Han, and J. Fang. Multi-dimensional Index on Hadoop Distributed File System. In *ICNAS*, July 2010.

[32] J. Lu and R. H. Güting. Parallel Secondo: Boosting Database Engines with Hadoop. In *ICPADS*, 2012.

[33] P. Lu et al. ScalaGiST: Scalable Generalized Search Trees for MapReduce Systems. *PVLDB*, 7(14), 2014.

[34] W. Lu, Y. Shen, S. Chen, and B. C. Ooi. Efficient Processing of k Nearest Neighbor Joins using MapReduce. *PVLDB*, 5(10), 2012.

[35] Q. Ma, B. Yang, W. Qian, and A. Zhou. Query Processing of Massive Trajectory Data Based on MapReduce. In *CLOUDDB*, 2009.

[36] A. Magdy et al. Taghreed: A System for Querying, Analyzing, and Visualizing Geotagged Microblogs. In *SIGSPATIAL*, 2014.

[37] H. Markram. The Blue Brain Project. *Nature Reviews Neuroscience*, 7(2), 2006.

[38] S. Nishimura et al. $\mathcal{MD}$-HBase: Design and Implementation of an Elastic Data Infrastructure for Cloud-scale Location Services. *DAPD*, 31(2), 2013.

[39] Open Geospatial Consortium, 2015.

[40] C. Olston et al. Pig Latin: A Not-so-foreign Language for Data Processing. In *SIGMOD*, 2008.

[41] J. Patel and D. DeWitt. Partition Based Spatial-Merge Join. In *SIGMOD*, June 1996.

[42] J. M. Patel et al. Building a Scaleable Geo-Spatial DBMS: Technology, Implementation, and Evaluation. In *SIGMOD*, 1997.

[43] L. W. Pickle et al. The Crossroads of GIS and Health Information: A Workshop on Developing a Research Agenda to Improve Cancer Control. *International Journal of Health Geographics*, 5(1), 2006.

[44] G. Planthaber et al. EarthDB: Scalable Analysis of MODIS Data using SciDB. In *BIGSPATIAL*, Nov. 2012.

[45] J. Sankaranarayanan, H. Samet, B. E. Teitler, and M. D. L. J. Sperling. TwitterStand: News in Tweets. In *SIGSPATIAL*, 2009.

[46] M. Stonebraker, P. Brown, A. Poliakov, and S. Raman. The Architecture of SciDB. In *SSDBM*, 2011.

[47] M. Stonebraker et al. SciDB: A Database Management System for Applications with Complex Analytics. *Computing in Science and Engineering*, 15(3), 2013.

[48] Y. F. Thomas, D. Richardson, and I. Cheung. *Geography and Drug Addiction*. Springer Verlag, 2009.

[49] A. Thusoo et al. Hive: A Warehousing Solution over a Map-Reduce Framework. *PVLDB*, 2009.

[50] J. VanderPlas et al. Squeezing a Big Orange into Little Boxes: The AscotDB System for Parallel Processing of Data on a Sphere. *IEEE Data Engineering Bulletin*, 36(4), 2013.

[51] H. T. Vo et al. Parallel Visualization on Large Clusters using MapReduce. In *LDAV*, 2011.

[52] G. Wang et al. Behavioral Simulations in MapReduce. *PVLDB*, 2010.

[53] K. Wang et al. Accelerating Spatial Data Processing with MapReduce. In *ICPADS*, 2010.

[54] R. T. Whitman, M. B. Park, S. A. Ambrose, and E. G. Hoel. Spatial Indexing and Analytics on Hadoop. In *SIGSPATIAL*, 2014.

[55] E. Wu, L. Battle, and S. R. Madden. The case for data visualization management systems. *PVLDB*, 7(10), 2014.

[56] D. Xie, F. Li, B. Yao, G. Li, L. Zhou, and M. Guo. Simba: Efficient In-Memory Spatial Analytics. In *SIGMOD*, June 2016.

[57] S. You, J. Zhang, and L. Gruenwald. Large-scale spatial join query processing in Cloud. In *CLOUDDM*, Apr. 2015.

[58] J. Yu et al. GeoSpark: A Cluster Computing Framework for Processing Large-Scale Spatial Data. In *SIGSPATIAL*, Nov. 2015.

[59] M. Zaharia et al. Spark: Cluster Computing with Working Sets. In *HotCloud*, 2010.

[60] C. Zhang, F. Li, and J. Jestes. Efficient Parallel kNN Joins for Large Data in MapReduce. In *EDBT*, Mar. 2012.

[61] S. Zhang, J. Han, Z. Liu, K. Wang, and S. Feng. Spatial Queries Evaluation with MapReduce. In *GCC*, pages 287–292, 2009.

[62] S. Zhang, J. Han, Z. Liu, K. Wang, and Z. Xu. SJMR: Parallelizing spatial join with MapReduce on clusters. In *CLUSTER*, Aug. 2009.

[63] X. Zhang et al. An Efficient Multi-Dimensional Index for Cloud Data Management. In *CLOUDDB*, 2009.

[64] W. Zhao, H. Ma, and Q. He. Parallel K-Means Clustering Based on MapReduce. In *CLOUDCOM*, 2009.