Geospatial Data Management in Apache Spark: A Tutorial

Jia Yu [1], Mohamed Sarwat [2]

School of Computing, Infomatics, and Decision Systems Engineering
Arizona State University
699 S Mill Avenue, Tempe, AZ 85281

[1] jiayu2@asu.edu

[3] msarwat@asu.edu

***Abstract*—The volume of spatial data increases at a staggering rate. This tutorial comprehensively studies how existing works extend Apache Spark to uphold massive-scale spatial data. During this 1.5 hour tutorial, we first provide a background introduction of the characteristics of spatial data and the history of distributed data management systems. A follow-up section presents the common approaches used by the practitioners to extend Spark and introduces the vital components in a generic spatial data management system. The third, fourth and fifth sections then discuss the ongoing efforts and experience in spatial-temporal data, spatial data analytics and streaming spatial data, respectively. The sixth part finally concludes this tutorial to help the audience better grasp the overall content and points out future research directions.**

I. Introduction

The volume of spatial data increases at a staggering rate. Such data includes earth science datasets, geotagged social media, vehicle trajectories, and sensor measurements. Furthermore, everything we do on our mobile and wearable devices, e.g., booking a taxi trip or making a dinner reservation, leaves breadcrumbs of geospatial digital traces. Existing relational DBMSs [22] support a variety of spatial data types, operators and index structures to process spatial operations but most of them fail at scaling up. To tackle this issue and scale out spatial operations, recent works, such as SpatialHadoop [8] and HadoopGIS [1] for Hadoop MapReduce, have harnessed distributed data management systems. Although these approaches achieve high scalability, they still exhibit slow run time performance and the user will not tolerate such delays. Apache Spark, on the other hand, provides a novel in-memory data abstraction called Resilient Distributed Datasets (RDDs) [38] to outperform existing models. Unfortunately, the native Spark ecosystem does not offer spatial data types and operations. Hence, there is a large body of research focusing on extending Spark to handle spatial data, indexes and queries.

This tutorial is expected to deliver a comprehensive study of how existing works incorporate Spark to uphold massive-scale spatial data. We also want this tutorial to serve as an introductory course that teaches the audience the basic building blocks in a scalable spatial data management system and the important design concerns based on our previous experience [34], [35], [36], [37]. Several other systems in Hadoop [10] and Flink [3] are also included to point out potential research directions for Spark-based systems. We begin our tutorial with a background introduction of the characteristics of spatial data and the history of distributed data management systems. A follow-up section presents common approaches used by the practitioners to extend Spark and introduces the vital components in a generic spatial data management system. The third, fourth and fifth sections then discuss the challenges and ongoing efforts in spatial-temporal data, spatial data analytics and streaming spatial data, respectively. The sixth part finally concludes this tutorial and points out future research directions.

1. **Big geospatial data** (10 mins.)
   1.1. The emergence of big spatial data
   1.2. A survery of Spatial data support in distributed data management systems
2. **Managing spatial data in Spark** (20 mins.)
   2.1. Extending the Spark API to support spatial operations
   2.2. Spatial index structures
   2.3. Spatial query processing
3. **Managing Spatio-Temporal Data in Spark** (15 mins.)
   3.1. Spatial-temporal indexes and partitioning methods
   3.2. Spatial-temporal query processing
   3.3. Processing Trajectory data in Spark
4. **Geospatial Data Analytics in Spark** (15 mins.)
   4.1. Geospatial visual analytics
   4.2. Geospatial statistical analysis
   4.3. Geospatial data mining and machine learning
5. **Processing streaming spatial data in Spark** (15 mins.)
   5.1. Concepts and challenges
   5.2. Streaming spatial query processing
6. **Wrap Up and Future directions** (15 minutes)

Fig. 1: Tutorial outline, 1.5 hour

II. Tutorial outline

Figure 1 depicts the outline of this tutorial which consists of six sections. During this 1.5-hour tutorial, we first motivate the idea, then start from the generic spatial data systems on Spark,

TABLE I: Geospatial data management systems in Apache Spark

	Spatial data type	Approach	Spatial indexing	Queries	Optimization	Temporal attribute	Streaming processing
GeoSpark [34], [35], [37]	Generic	RDD, DataFrame	Two-level	Range, Join, KNN	Query optimizer, object serializer	Not optimized	Not optimized
Simba [32]	Generic	DataFrame	Two-level	Range, Join, KNN, KNN join	Query optimizer	Not optimized	Not optimized
LocationSpark [29]	Generic	DataFrame	Two-level	Range, Join, KNN, KNN join	Query optimizer	Not optimized	Not optimized
GeoMesa [12]	Generic	RDD, DataFrame	Global grid file	Range,Join	-	Not optimized	Not optimized
Magellan [17]	Generic	DataFrame	-	Range,Join	-	Not optimized	Not optimized
SpatialSpark [33]	Generic	RDD	Two-level	Range, Join	-	-	-
SparkGIS [7]	Generic	RDD	Two-level	Range, Join, KNN	Resource-aware query rewriter	-	-
DST [31]	Trajectory	DataFrame	Two-level	Similarity search	-	Not optimized	Not optimized
DITA [27]	Trajectory	DataFrame	Two-level	Similarity join	Query optimizer	Not optimized	Not optimized
SciSpark [20]	Satellite image	RDD	-	Filter, Join	-	Not optimized	-
GeoSparkViz [36]	Raster map	RDD	-	Range, Join, Overlay	-	-	-
Geotrellis [14]	Raster map	RDD	-	Cropping, Warping, Map algebra	-	Not optimized	-
BinJoin [30]	Generic	RDD	Local index	Join	Query optimizer	Optimized	-

and elaborate systems for specific spatial data and present some real-world examples. All reviewed Spark-based systems are listed in Table I.

A. Section 1: The overview of big spatial data and Spark

The first section takes 10 minutes, as shown in Figure 1. We initiate our tutorial with real life spatial data use cases and further explain the recent explosion of big spatial data. We then go through spatial data support in existing distributed data management systems, including MPI-GIS [24], Parallel Secondo [16], HadoopGIS [1], SpatialHadoop [8], ESRI tools for Hadoop [9], Presto-Spatial [23], and MD-HBase [19]. Additionally, we illustrate some important concepts in Apache Spark such as Resilient Distributed Dataset (RDD) [38] and SQL [5] to explain why Spark outperforms state-of-art systems. Finally, we plan to show the performance differences between the existing systems and Spark when used to perform classic spatial operations (e.g., spatial range query and join query) as reported in recent literature [32], [37], [21].

B. Section 2: Managing spatial data in Spark

The second section costs around 20 minutes. In this part, we first explore the common approaches that are used to extend Apache Spark for supporting generic spatial data. For the ease of understanding, we put the existing approaches into two categories, RDD-based and DataFrame-based, according to the Spark components they connect. The RDD-based approaches such as LocationSpark [29] and SpatialSpark [33] directly extend bare metal RDD in Spark and allow the users to gain granular control over spatial operation execution plan. DataFrame-based approaches such as Simba [32] and Magellan [17] extend SparkSQL catalyst with customized spatial query optimization. This approach hides the internal query execution and allows users to draw declarative queries. Some systems (e.g., GeoSpark [37] and GeoMesa [12]) provide Spatial SQL interfaces [6], [11] besides RDD and DataFrame.

We will then go through the basic components that play important roles in building spatial data management systems in Spark. We first describe how distributed spatial indexing is done in Spark. The existing systems [32], [37], [29] generally build two-level index structures: a global succinct index with local tree indexes on each RDD partition. Second, we will explore the techniques used to accelerate spatial query processing in Spark. For example, GeoSpark [37] leverages KDB-tree based spatial partitioning technique to avoid time-consuming Spark default join mechanism while Simba [32] and GeoMesa [12] use R-Tree partitioning instead. LocationSpark [29] and Simba [32] support K Nearest Neighbor-Join query which is totally not supported in Spark. These systems also possess query optimizers to yield efficient execution plans.

In addition, we present some other components that are critical for running such in-memory spatial data systems. For instance, the customized spatial object serializer in GeoSpark [37] compresses loose in-memory spatial objects and indexes to dramatically decrease memory footprint of spatial operations. Resource-aware query rewriter in SparkGIS [7] can automatically rewrite a spatial query that exceeds the memory limitation of a Spark cluster to a batch of smaller queries each of which can fully run in memory.

C. Section 3: Managing Spatial-Temporal Data in Spark

As given in Figure 1, the third section takes 15 minutes to present research efforts on taming spatial-temporal data in Spark. We start by explaining the characteristics of spatial-

temporal data and reveal the challenges [15]. This leads us to describe the limitations of some existing work in Spark [12], [31], [37]: their index structures or data partitioning techniques do not take into account the temporal attribute so they can only treat a temporal query predicate (e.g., find spatial objects occur within a time interval) as a simple data filter.

We then explore several ongoing efforts [30], [2], [28] on incorporating temporal attributes, including Bin join [30] which partitions spatial data in Spark according to spatial and temporal proximity. Furthermore, we also review some works in Hadoop MapReduce framework, such as ST-Hadoop [2] which supports multi-layer spatial-temporal indexes and CloST [28] which partition spatial data based on temporal proximity, to point out possible research directions.

Another important type of spatial-temporal data is trajectories. We present the limitations of generic spatial data systems when performing queries on trajectories, such as extensive overlapped spatial boundaries of RDD partitions, inaccurate distance metrics and inefficient index structures. We will also describe existing efforts for segment-oriented trajectory partition methods (e.g., DFT [31]), Hausdorff/Frechet/DTW distance metrics (e.g., DFT [31] and DITA [27]) and novel trajectory indexes (e.g., DITA [27]).

D. Section 4: Geospatial Data Analytics in Spark

The fourth section will take another 15 minutes to demonstrate how to perform geospatial analytics in Spark using these spatial data systems. We will first go through the works (e.g., SciSpark [20], GeoSparkViz [36] and GeoTrellis [14]) on extending Spark to support spatial visual analytics which generally produce raster map image and satellite image. To be precise, each RDD partition in such systems is a self-contained 2D array dataset that includes some meta information describing the time and spatial location of this partition. We will explain how this makes their system paradigms different from other spatial data management systems in Spark and showcase visual analytics examples including global climate changes and traffic distribution.

Furthermore, we will direct the audience to run spatial statistical analytics, such as spatial aggregation analytics and spatial hot spot analysis, using one of the aforementioned big spatial data systems (e.g., GeoSpark [37]). In addition, we will also give two case studies about how to perform spatial data mining and machine learning in Spark: (1) spatial regression analysis (2) spatial co-location pattern mining. These real-world examples are expected to help the audience better understand how to apply these techniques to their research.

E. Section 5: Processing streaming spatial data in Spark

We will take another 15 minutes in Section 5 to discuss the recent work of processing spatial streaming data in Spark. The first part of this section will describe the structured streaming in Spark [4] which provides a declarative DataFrame SQL API to users. This part will further show how the streaming component differs from the regular Spark RDD and DataFrame API. We then describe the mechanism of directly applying the existing generic Spark spatial systems to streaming applications [37], [12]. We will explain why these systems cannot yield the best performance although they can more or less work with spatial streaming data.

Furthermore, we review a couple of ongoing efforts in other relevant ecosystems such as Apache Flink [3] and Microsoft SQL Server [18] to convey some insights of building a distributed or centralized spatial streaming system. We hope that this can guide practitioners to develop better systems in Spark environment. This will cover some important topics including how to perform incremental spatial analysis and partition spatial data [26], event prediction [25] and query refinement [13].

F. Wrap up and future directions

In the final session of the tutorial which lasts for 15 minutes, we conclude the changes made by recent trends in extending Spark to support spatial data and also introduce several important future research directions. Specifically, we reveal two directions: (1) Exploring specific efficient algorithms and data structures to process spatial-temporal and spatial streaming data in Spark. (2) Developing efficient query optimization strategies and Spark-specific optimization components to help Spark Catalyst optimizer generate better distributed spatial query execution plans.

III. Intended audience and duration

The tutorial bridges the gap between two broad areas that are deemed quite necessary in the data science stack: (1) Distributed in-memory computation engine Spark and (2) Big spatial data. Hence, our tutorial targets mainly data scientists, data management researchers / practitioners, and data enthusiasts. The tutorial lasts for 1.5 hours (detailed timing is given in Figures 1) and attending the tutorial does not require any prior knowledge as it starts by giving a quick overview of distributed data systems and big spatial data. By attending the tutorial, the audience is expected to learn about cutting-edge spatial data management techniques in Spark and get more familiar with the state-of-the-art research (i.e., systems, tools, applications) that lies in the intersection of both database systems and GIS. More specifically, data scientists will learn how to tweak existing Spark-based systems in the data science stack (i.e., spatial query execution and spatial data mining) to minimize the data-to-insight time over massive-scale data. Database researchers will benefit from the tutorial since the presenters will describe a set of future research directions in distributed spatial data management systems.

IV. Speaker biographies

Jia Yu is a Ph.D. student at the Computer Science department, School of Computing, Informatics, and Decision Systems Engineering, Arizona State University. Jia's research focuses on database systems and geospatial data management. In particular, he worked on distributed data management systems, database indexing, data visualization, code generation with JIT execution. He is the main contributor of several open-sourced research projects such as GeoSpark [1], one of

[1]GeoSpark website: http://datasystemslab.github.io/GeoSpark

the de-facto spatial data management frameworks in Spark ecosystem.

Mohamed Sarwat is an Assistant Professor of Computer Science and the director of the Data Systems lab at Arizona State University. Before joining ASU, Mohamed obtained his PhD degree in computer science from University of Minnesota in 2014. His research interest lies in the broad area of data management systems. Mohamed is a recipient of the University of Minnesota doctoral dissertation fellowship. His research work has been recognized by two best research paper awards in MDM 2015 and SSTD 2011 as well as a Best of Conference citation in ICDE 2012. He also received CCC Blue Sky Ideas award for best vision papers (3rd place) in SSTD 2017. Mohamed is an associate editor for the GeoInformatica journal and has served as a PC member for major data management and spatial computing venues.

REFERENCES

[1] AJI, A., WANG, F., VO, H., LEE, R., LIU, Q., ZHANG, X., AND SALTZ, J. H. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce. *Proceedings of the VLDB Endowment, PVLDB 6*, 11 (2013), 1009–1020.

[2] ALARABI, L., MOKBEL, M. F., AND MUSLEH, M. St-hadoop: A mapreduce framework for spatio-temporal data. In *Proceedings of the International Symposium on Advances in Spatial and Temporal Databases, SSTD* (2017), pp. 84–104.

[3] APACHE. FLINK. https://flink.apache.org/.

[4] ARMBRUST, M., DAS, T., TORRES, J., YAVUZ, B., ZHU, S., XIN, R., GHODSI, A., STOICA, I., AND ZAHARIA, M. Structured streaming: A declarative API for real-time applications in apache spark. In *Proceedings of the ACM International Conference on Management of Data, SIGMOD* (2018), pp. 601–613.

[5] ARMBRUST, M., XIN, R. S., LIAN, C., HUAI, Y., LIU, D., BRADLEY, J. K., MENG, X., KAFTAN, T., FRANKLIN, M. J., GHODSI, A., AND ZAHARIA, M. Spark SQL: relational data processing in spark. In *Proceedings of the ACM International Conference on Management of Data, SIGMOD* (2015), pp. 1383–1394.

[6] ASHWORTH, M. Information technology – database languages – sql multimedia and application packages – part 3: Spatial. Standard, International Organization for Standardization, Geneva, Switzerland, 2016.

[7] BAIG, F., VO, H., KURÇ, T. M., SALTZ, J. H., AND WANG, F. Sparkgis: Resource aware efficient in-memory spatial query processing. In *International Conference on Advances in Geographic Information Systems* (2017), pp. 28:1–28:10.

[8] ELDAWY, A., AND MOKBEL, M. F. Spatialhadoop: A mapreduce framework for spatial data. In *Proceedings of the International Conference on Data Engineering, ICDE* (2015), pp. 1352–1363.

[9] ESRI, GIS. Tools for Hadoop, 2015.

[10] Apache Hadoop. http://hadoop.apache.org/.

[11] HERRING, J. R. Opengis implementation specification for geographic information-simple feature access-part 2: Sql option. *Open Geospatial Consortium Inc* (2006).

[12] HUGHES, J. N., ANNEX, A., EICHELBERGER, C. N., FOX, A., HULBERT, A., AND RONQUEST, M. Geomesa: a distributed architecture for spatio-temporal fusion. In *Geospatial Informatics, Fusion, and Motion Video Analytics V* (2015), vol. 9473, International Society for Optics and Photonics, p. 94730F.

[13] JIANG, D., OOI, B. C., SHI, L., AND WU, S. The Performance of MapReduce: An in-depth Study. *Proceedings of the International Conference on Very Large Data Bases, VLDB 3*, 1-2 (2010), 472–483.

[14] KINI, A., AND EMANUELE, R. Geotrellis: Adding geospatial capabilities to spark. *Spark Summit* (2014).

[15] LI, Z., HU, F., SCHNASE, J. L., DUFFY, D. Q., LEE, T., BOWEN, M. K., AND YANG, C. A spatiotemporal indexing approach for efficient processing of big array-based climate data with mapreduce. *International Journal of Geographical Information Science 31*, 1 (2017).

[16] LU, J., AND GUTING, R. H. Parallel Secondo: Boosting Database Engines with Hadoop. In *International Conference on Parallel and Distributed Systems* (2012), pp. 738 –743.

[17] MAGELLAN: GEO SPATIAL DATA ANALYTICS ON SPARK. https://github.com/harsha2010/magellan.

[18] MICROSOFT SQL SERVER. https://www.microsoft.com/en-us/sql-server/default.aspx.

[19] NISHIMURA, S., DAS, S., AGRAWAL, D., AND ABBADI, A. E. MD-Hbase: A Scalable Multi-dimensional Data Infrastructure for Location Aware Services. In *Proceedings of the International Conference on Mobile Data Management, MDM* (2011), pp. 7–16.

[20] PALAMUTTAM, R., MOGROVEJO, R. M., MATTMANN, C., WILSON, B., WHITEHALL, K., VERMA, R., MCGIBBNEY, L. J., AND RAMIREZ, P. M. Scispark: Applying in-memory distributed computing to weather event detection and tracking. In *2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, October 29 - November 1, 2015* (2015), pp. 2020–2026.

[21] PANDEY, V., KIPF, A., NEUMANN, T., AND KEMPER, A. How good are modern spatial analytics systems? *Proceedings of the VLDB Endowment, PVLDB 11*, 11 (2018), 1661–1673.

[22] PostGIS. http://postgis.net.

[23] Presto. https://prestodb.io/.

[24] PURI, S., AND PRASAD, S. K. Mpi-gis: New parallel overlay algorithm and system prototype.

[25] QADAH, E., MOCK, M., ALEVIZOS, E., AND FUCHS, G. A distributed online learning approach for pattern prediction over movement event streams with apache flink. In *Proceedings of the Workshops of the EDBT/ICDT 2018 Joint Conference (EDBT/ICDT 2018), Vienna, Austria, March 26, 2018.* (2018), pp. 109–116.

[26] SALMON, L., AND RAY, C. Design principles of a stream-based framework for mobility analysis. *GeoInformatica 21*, 2 (2017), 237–261.

[27] SHANG, Z., LI, G., AND BAO, Z. DITA: distributed in-memory trajectory analytics. In *Proceedings of the ACM International Conference on Management of Data, SIGMOD* (2018), pp. 725–740.

[28] TAN, H., LUO, W., AND NI, L. M. Clost: a hadoop-based storage system for big spatio-temporal data analytics. In *Proceedings of the International Conference on Information and Knowledge Management, CIKM* (2012), pp. 2139–2143.

[29] TANG, M., YU, Y., MALLUHI, Q. M., OUZZANI, M., AND AREF, W. G. LocationSpark: A Distributed In-Memory Data Management System for Big Spatial Data. *Proceedings of the VLDB Endowment, PVLDB 9*, 13 (2016), 1565–1568.

[30] WHITMAN, R. T., PARK, M. B., MARSH, B. G., AND HOEL, E. G. Spatio-temporal join on apache spark. In *International Conference on Advances in Geographic Information Systems* (2017), pp. 20:1–20:10.

[31] XIE, D., LI, F., AND PHILLIPS, J. M. Distributed trajectory similarity search. *Proceedings of the VLDB Endowment, PVLDB 10*, 11 (2017), 1478–1489.

[32] XIE, D., LI, F., YAO, B., LI, G., ZHOU, L., AND GUO, M. Simba: Efficient In-Memory Spatial Analytics. In *Proceedings of the ACM International Conference on Management of Data, SIGMOD* (2016).

[33] YOU, S., ZHANG, J., AND GRUENWALD, L. Large-scale spatial join query processing in cloud. In *31st IEEE International Conference on Data Engineering Workshops, ICDE Workshops 2015, Seoul, South Korea, April 13-17, 2015* (2015), pp. 34–41.

[34] YU, J., WU, J., AND SARWAT, M. GeoSpark: A Cluster Computing Framework for Processing Large-Scale Spatial Data. In *Proceedings of the ACM Symposium on Advances in Geographic Information Systems, SIGSPATIAL* (2015).

[35] YU, J., WU, J., AND SARWAT, M. A demonstration of geospark: A cluster computing framework for processing big spatial data. In *Proceedings of the International Conference on Data Engineering, ICDE* (2016), pp. 1410–1413.

[36] YU, J., ZHANG, Z., AND SARWAT, M. Geosparkviz: a scalable geospatial data visualization framework in the apache spark ecosystem. In *Proceedings of the International Conference on Scientific and Statistical Database Management, SSDBM* (2018), pp. 15:1–15:12.

[37] YU, J., ZHANG, Z., AND SARWAT, M. Spatial data management in apache spark: The geospark perspective and beyond. *Geoinformatica* (2018).

[38] ZAHARIA, M., CHOWDHURY, M., DAS, T., DAVE, A., MA, J., MCCAULY, M., FRANKLIN, M. J., SHENKER, S., AND STOICA, I. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation, NSDI* (2012), pp. 15–28.