Spatial Data Wrangling with GeoSpark - A Step by Step Tutorial

Jia Yu
Arizona State University, Tempe AZ
jiayu2@asu.edu

Mohamed Sarwat
Arizona State University, Tempe AZ
msarwat@asu.edu

ABSTRACT

This tutorial is expected to deliver a comprehensive study and hands-on tutorial of how GeoSpark incorporates 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. 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 section gives a hands-on live demonstration to illustrate the basic steps of performing geospatial data analytics using GeoSpark.

CCS CONCEPTS

• **Information systems** → *Spatial-temporal systems*; • **Computing methodologies** → *Distributed algorithms.*

KEYWORDS

Cluster computing; Large-scale data; Spatial data

ACM Reference Format:
Jia Yu and Mohamed Sarwat. 2019. Spatial Data Wrangling with GeoSpark - A Step by Step Tutorial. In *1st ACM SIGSPATIAL International Workshop on Geospatial Data Access and Processing APIs (SpatialAPI'19), November 5, 2019, Chicago, IL, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3356394.3365589

1 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 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 [4] 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) [14] 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 and hands-on tutorial of how GeoSpark [10–13] incorporates 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. 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 section gives a hands-on live demonstration to illustrate the basic steps of performing geospatial data analytics using GeoSpark.

(1) **Big geospatial data** (20 mins.)
  (a) The emergence of big spatial data
  (b) A survery of Spatial data support in distributed data management systems
(2) **Managing spatial data in Spark** (50 mins.)
  (a) Extending the Spark API to support spatial operations
  (b) Spatial index structures
  (c) Spatial query processing
(3) **Geospatial Data Analytics in Spark** (50 mins.)
  (a) Geospatial visual analytics
  (b) Geospatial statistical analysis
  (c) Geospatial data mining and machine learning

Figure 1: Tutorial outline, 2 hours

2 TUTORIAL OUTLINE

2.1 Sec. 1: Big spatial data in Spark

The first section takes 20 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 HadoopGIS [1], SpatialHadoop [4], ESRI tools for Hadoop [5], and MD-HBase [7]. Additionally, we illustrate some important concepts in Apache Spark such as Resilient Distributed Dataset (RDD) [14] and SQL [2] 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.

SpatialAPI'19, November 5, 2019, Chicago, IL, USA

ACM ISBN 978-1-4503-6953-4/19/11…$15.00
https://doi.org/10.1145/3356394.3365589

2.2 Sec. 2: Managing spatial data in Spark

The second section costs around 50 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 directly extend bare metal RDD in Spark and allow the users to gain granular control over spatial operation execution plan. DataFrame-based approaches 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 [13]) provide Spatial SQL interfaces 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 [8, 9, 13] 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 [13] leverages KDB-tree based spatial partitioning technique to avoid time-consuming Spark default join mechanism while Simba [9] and GeoMesa [6] use R-Tree partitioning instead. LocationSpark [8] and Simba [9] 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 [13] compresses loose in-memory spatial objects and indexes to dramatically decrease memory footprint of spatial operations. Resource-aware query rewriter in SparkGIS [3] 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.

2.3 Sec. 3: Spatial Data Analytics in GeoSpark

This section will take another 50 minutes to demonstrate how to perform geospatial analytics in Spark using GeoSpark. We will first go through GeoSparkViz which extends Spark to support spatial visual analytics. It generally produces 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 GeoSpark. 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.

3 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 2 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 GeoSpark 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. **We will provide a virtual machine file which includes Spark, GeoSpark and Zeppelin. Attendees who are interested in the hands-on demonstration should have Virtual Box installed.**

4 ACKNOWLEDGMENT

This work is supported by the National Science Foundation (NSF) under Grant 1845789.

REFERENCES

[1] Ablimit Aji, Fusheng Wang, Hoang Vo, Rubao Lee, Qiaoling Liu, Xiaodong Zhang, and Joel H. Saltz. 2013. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce. *PVLDB* 6, 11 (2013), 1009–1020.
[2] Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley, Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, and Matei Zaharia. 2015. Spark SQL: Relational Data Processing in Spark. In *SIGMOD*. 1383–1394.
[3] Furqan Baig, Hoang Vo, Tahsin M. Kurç, Joel H. Saltz, and Fusheng Wang. 2017. SparkGIS: Resource Aware Efficient In-Memory Spatial Query Processing. In *SIGSPATIAL*. 28:1–28:10.
[4] Ahmed Eldawy and Mohamed F. Mokbel. 2015. SpatialHadoop: A MapReduce framework for spatial data. In *ICDE*. 1352–1363.
[5] Esri, GIS. 2015. Tools for Hadoop.
[6] James N Hughes, Andrew Annex, Christopher N Eichelberger, Anthony Fox, Andrew Hulbert, and Michael Ronquest. 2015. Geomesa: a distributed architecture for spatio-temporal fusion. In *Geospatial Informatics, Fusion, and Motion Video Analytics V*, Vol. 9473. International Society for Optics and Photonics, 94730F.
[7] Shoji Nishimura, Sudipto Das, Divyakant Agrawal, and Amr El Abbadi. 2011. MD-Hbase: A Scalable Multi-dimensional Data Infrastructure for Location Aware Services. In *MDM* (June). 7–16.
[8] MingJie Tang, Yongyang Yu, Qutaibah M. Malluhi, Mourad Ouzzani, and Walid G. Aref. 2016. LocationSpark: A Distributed In-Memory Data Management System for Big Spatial Data. *PVLDB* 9, 13 (2016), 1565–1568.
[9] Dong Xie, Feifei Li, Bin Yao, Gefei Li, Liang Zhou, and Minyi Guo. 2016. Simba: Efficient In-Memory Spatial Analytics. In *SIGMOD*.
[10] Jia Yu, Jinxuan Wu, and Mohamed Sarwat. 2015. GeoSpark: A Cluster Computing Framework for Processing Large-Scale Spatial Data. In *SIGSPATIAL*.
[11] Jia Yu, Jinxuan Wu, and Mohamed Sarwat. 2016. A demonstration of GeoSpark: A cluster computing framework for processing big spatial data. In *ICDE*. 1410–1413.
[12] Jia Yu, Zongsi Zhang, and Mohamed Sarwat. 2018. GeoSparkViz: a scalable geospatial data visualization framework in the apache spark ecosystem. In *SSDBM*. 15:1–15:12. https://doi.org/10.1145/3221269.3223040
[13] Jia Yu, Zongsi Zhang, and Mohamed Sarwat. 2018. Spatial Data Management in Apache Spark: The GeoSpark Perspective and Beyond. *Geoinformatica* (2018).
[14] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauly, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2012. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. In *NSDI*. 15–28.