Framework for Visualization of GeoSpatial Query Processing by Integrating Redis With Spark

S. Vasavi, V.R. Siddhartha Engineering College, Vijayawada, India
V. N. Priyanka G, V.R. Siddhartha Engineering College, Vijayawada, India
Anu A. Gokhale, Illinois State University, Normal, USA

ABSTRACT
Nowadays we are moving towards digitization and making all our devices produce a variety of data, this has paved the way to the emergence of NoSQL databases like Cassandra, NoSQL, and Redis. Big data such as geospatial data allows for geospatial analytics in applications such as tourism, marketing, and rural development. Spark frameworks provide operators storage and processing of distributed data. This article proposes “GeoRediSpark” to integrate Redis with Spark. Redis is a key-value store that uses an in-memory store, hence integrating Redis with Spark can extend the real-time processing of geospatial data. The article investigates storage and retrieval of the Redis built-in geospatial queries and has added two new geospatial operators, GeoWithin and GeoIntersect, to enhance the capabilities of Redis. Hashed indexing is used to improve the processing performance. A comparison on Redis metrics with three benchmark datasets is made. Hashset is used to display geographic data. The output of geospatial queries is visualized to the type of place and the nature of the query using Tableau.

KEYWORDS
Consistent Hashing, GeoHash, Geospatial Data, Hashset, Location-Based Query, Master- Executor Daemon, Remote Dictionary Server (Redis) Metrics, Resilient Distributed Datasets (RDD)

1. INTRODUCTION
Companies that use big data for business challenges can gain advantage by integrating Redis with Spark. Spark framework provides support for analytics, where process execution is fast because of in-memory optimization. Out of various NoSQL databases, Redis provides key-value pair, in-memory storage and suits to applications that require fast results. As such, when integrated, Redis and Spark together can index data efficiently and helps in analytics of variety of data driven applications. Geospatial data helps in identifying the geographic location of an object, its features and boundaries on Earth. Such data can be analyzed to serve various purposes such as tourism, health care, geo marketing and intelligent transportation system. There are two data types of spatial data, vector and raster. Both data types stores object reference as latitude and longitude (vertices/paths or grid cells). Raster data

DOI: 10.4018/IJNCR.2019070101

Copyright © 2019, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
includes remote sensing, photogrammetric, and vector data includes Geographical Positioning System (GPS) data. Raster data can be represented in its original resolution and form without generalization. But the location of each vertex needs to be stored explicitly. Advantage of vector data is that, geographic location of each cell is implied by its position in the cell matrix. The disadvantage being, it is difficult to adequately represent linear features depending on the cell resolution.

Tableau uses various file formats such as KML, ERSI shape files, GeoJSON files, MapInfo interchange formats for geographic data analysis and display. Traditional databases (relational database) are suitable for storing and querying structured data that guarantees ACID properties. With the emergence of the internet, large amounts of unstructured data is being produced. NoSQL databases, that guarantees CAP properties are suitable for storing such unstructured data. Dynamo, Redis, MongoDB, BigTable, HBase, Cassandra are designed to handle the data storage and processing with less response time. Redis suits for complex queries such as social networking applications, where we have to optimise latency. Redis work with client and server in the same or on different systems. Redis server takes care of data management while client has programming language API. Master and slaves will take care about replication of data. As stated in (Ramel, 2016), for time series data analytics, Redis can speed up processing time.

Even though Redis has no declarative query language support, data can be indexed like in relational databases and structured as JSON fragments. Cassandra monitor nodes handles redundancy and can avoid lazy nodes, whereas Redis can monitor these activities at higher granular level. Even though some works are reported for labelling and retrieving Redis data, are not efficient either at indexing or at retrieval. This paper aims at adding the functionality of spatial querying for Redis database by integrating it with Spark.

Geospatial functions include zooming and panning, reordering layers, and selecting features. Most commonly used operations are to find the nearest locations of a specified source location. But finding these locations based on latitude-longitude coordinate values are really a bit difficult task, especially when dealing with high precision values. Geohashing technique can be used to overcome this problem. It takes latitude-longitude pair as input and produces a geohash value, whose length is based on precision value specified. Another major pitfall is, searching the entire database sequentially for a required destination using this geohash value may not deliver efficient results as expected. Thus, parallel processing must be done to get rid of this issue. To achieve this, Redis is integrated with Spark which is an efficient distributed parallel processing paradigm.

Spark integrated with NoSQL databases will take the advantage of schema flexibility, scalability and support to variety of data types required for data stream applications. Redis can be integrated with Spark either by using connector as shown in Figure 1 or by using Redis Java client: Jedis.

Figure 1. Integrating Redis with Spark using Redis connector (Foulger, 2016)
Classification of Faults in Power Transmission Systems Using Modern Techniques: An Overview
www.igi-global.com/chapter/classification-of-faults-in-power-transmission-systems-using-modern-techniques/177987?camid=4v1a

An Autonomous Multi-Agent Simulation Model for Acute Inflammatory Response
John Wu, David Ben-Arieh and Zhenzhen Shi (2011). International Journal of Artificial Life Research (pp. 105-121).
www.igi-global.com/article/autonomous-multi-agent-simulation-model/54751?camid=4v1a