A Big Spatiotemporal Analytics Engine for Air Traffic Movement data

Shaohua Wang[1], Ershun Zhong[2], Hao Lu[2], Weiying Yun[2], Wenwen Cai[2]

[1]Department of Geography, University of California, Santa Barbara, CA, 93117, USA
Email: shaohuawang@ucsb.edu.

[2]SuperMap Software Co., Ltd., Beijing 100015, China
Email: zhonges@supermap.com, luhao@supermap.com, yunweiying@supermap.com,caiwenwen@supermap.com

1. Introduction

Global air traffic has surpassed the 7 billion passenger mark in 2015, and expected to double by 2029 based on a projected yearly growth rate of 5.2% (Airports Council International, 2018). The increasing air flight demand aggravates air traffic congestions, causing billions of dollars' losses to passengers, airlines and airports each year.

A crucial aspect of solving the problem is to efficient understand at any time the air transport condition for example the flight records statistics at a specific airport within a certain temporal interval, or the occurring global air traffic congestion given certain spatial range, based on air traffic movement big data. This relates to computing-intensive tasks of spatiotemporal join operation.

Apache Spark is a widespread adopted open-source general computing framework for distributed processing huge volume of data. But it cannot directly support spatiotemporal analytics for air traffic movement data since the lack of spatial indexes and complex spatiotemporal operations.

Previous studies have realized and offered spatial and temporal analytic ability for big data within Apache Spark. For example, GeoSpark (Yu, et al.2016) provides quadtree and r-tree indexes and two specific spatial join operations of overlap and intersect. LocationSpark (Tang, et al.2016) and SpatialSpark (You, et al.2015) support a broadcast spatial join operations but does not provide temporal join. STARK (Hagedorn, et al., 2017) supports three temporal join operators (e.g., Contains, ContainedBy, and Intersects). These solutions show high performance in spatial join or temporal join operations, but they do not support comprehensive spatiotemporal join operations well.

In this paper, we have introduced the spatiotemporal join operation and implemented a big spatiotemporal analytics engine within Apache Spark for air traffic movement data. The engine supports comprehensive spatiotemporal join operators. It also provides other spatial analysis functions which can be integrated used with spatiotemporal join operators, which can be used for complex spatiotemporal analytics for air traffic movement data to facilitate decision making.

2. Methods

2.1 Spatiotemporal join

Spatiotemporal join is defined as the feature matching of input feature datasets (e.g., points, lines, and polygons) according to the attribute, spatial or temporal characteristics, or a combination of those characteristics under specific join condition. Taking air traffic movement data analysis as an example, the spatiotemporal join can be used to analyse whether there is overlap of two airplanes in temporal dimension, for example, whether they take off at the same time, or arrive at the same time.

The spatiotemporal join involves spatial, temporal, and attribute associations. Specifically, the judgment of spatial association supports six spatial join operators including Contains, Intersects, Covers, Meets, Near, and Equals. The judgment of temporal association supports 14 types of temporal join operators, including Before, After, Equal, Meets, MetBy, Overlaps, OverlappedBy, During, Contains, Starts, StartedBy, Finishes, FinishedBy, and Near (Table 1). Attribute association is defined as attribute join, which supports Equals and Unequal

Table 1. Fourteen temporal join operators.

Operators	**Description**	**Diagram**	**Operators**	**Description**
Before	X is before Y	XXX__YYY	After	X is after Y
Equals	X is equivalent to Y	XXX YYY	—	—
Meets	X meets Y, and X is before Y	XXXYYY	MetBy	Y meets X, and Y is before X
Overlaps	X overlaps Y, and X is before Y	XXX __YYY	OverlappedBy	X overlaps Y, and Y is before X
During	X is during Y	XXX YYYYYY	Contain	X contains Y
Starts	X starts before Y	XXX YYYYYY	StartedBy	X starts by Y
Finishes	X finishes, then Y	___XXX YYYYYY	FinishedBy	X finishes by Y
Near	X is near Y	XXX(<D)YYY	—	—

The spatiotemporal join types include one-to-one mode and one-to-many mode. In one-to-one mode, attribute aggregation can be performed on multiple objects connected with the same object. And the aggregation conditions supports Max, Min, Average, Sum, Variance, and StdDeviation.

The diagram of spatiotemporal join is shown in Figure 1. For the temporal association analytics, the temporal interval of the target and source dataset is calculated first, and then the time grid is divided, and at last, the relationship of the elements in the same grid is determined. For the spatial association analytics, the data is divided by the spatial grid index first, and then the spatial and temporal association is connected on specific join condition. Taking equality relationship judgement as an example of the attribute association analytics, the join operator of Spark is used, and the grouping statistics is conducted if in one-to-many mode.

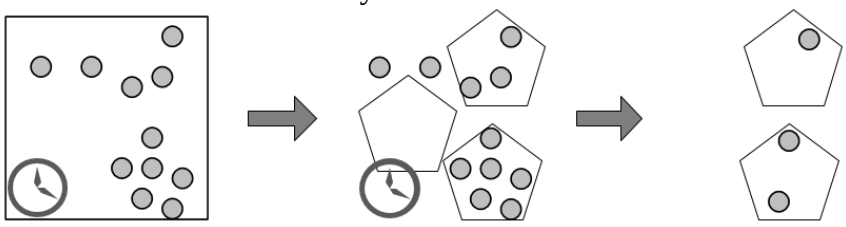


Figure 1. Diagram of the feature join given specific temporal interval and spatial range

2.2 Framework

The framework of the big spatiotemporal analytics engine is shown in Figure 2. Data cleaning is conducted first to reduce data uncertainty and the cleaned data can be stored in different distributed storage systems. Spatial data engine is used to read the cleaned data as FeatureRDD. The visualization of the analytics results can be viewed on different clients. These are not the focus of this paper and details are not described any more.

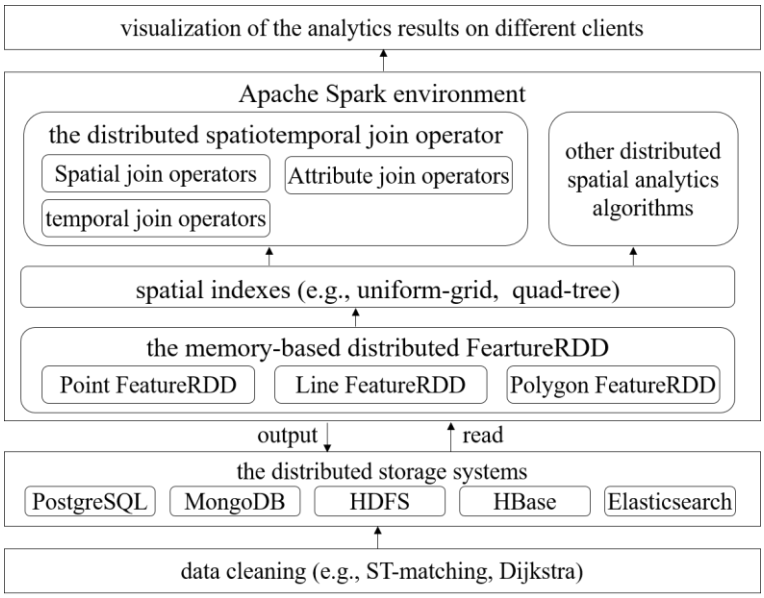


Figure 2. The framework of the big spatiotemporal analytics engine

The key part of the engine is illustrated in the middle of Figure 2. Spatial features expressed by points, lines and polygons, *et al.,* is realized using FeatureRDD (Wang et al., 2017). FeatureRDD is an extension of the memory-based distributed Resilient Distributed Dataset (RDD) of Apach Spark. The use of FeatureRDD facilitates processing spatiotemporal analytics natively in Apache Spark environment.

In the engine, spatial indexes like uniform-grid or quad-tree are implemented, which inherit from FeatureRDD. Spatiotemporal join algorithms including 6 spatial join operators, 14 temporal operators and attribute join operators are realized based on FeatureRDD (Figure 2). Other spatial analytics algorithms include spatial query, proximity analysis, buffer analysis, overlay analysis, aggregation analysis, density analysis, hot spot analysis and map matching are also realized (Figure 2).

3. Experiments

3.1 Data and environment

A 5 nodes Apache Spark cluster is constructed. Each node uses the 64-bit Ubuntu Linux operating system (16.04 version) and the Intel Core™ i7-6700K CPU processor (4 cores, master frequency of 4 GHz). The volume of the hard disk and memory of each node is 1TB, and 16 GB, respectively. SuperMap iObjects Java (which is an enterprise GIS software, and the big spatiotemporal analytics engine is part of the software) and BDT Spark extension package are installed and configured on each node. The Hadoop version is 2.7.3, and the Spark version is 2.1.0.

Our goal is to analyse the global flights congestion conditions in given spatial areas at specific time. Global single day air traffic movement data is used for benchmark experiments. And the join condition is 6 km distance interval in all spatial directions and 10 seconds temporal interval for any two airplanes. The dataset is 20.4GB and include over ten millions of air flight records. The data is stored as CSV format.

3.1 Detailed analysis process

Table 2 shows the command example of submitting the main function to perform the above analysis using ‘spark-submit’ command. The CSV file of the global flight data is read as RDD first, and the airplane on the ground are excluded according to

data attributes at the same time. The RDD dataset is the used as the destination dataset and the source dataset. The one-to-many mode is used for feature matching, and the results are summarized. If interested, one-to-one mode can be used to analyse and statistic the number of early warning airplanes (i.e., airplanes near to a particular airport or a specific airplane.

Information of flight number, spatial location, flight time of each airplane are retained for participating spatiotemporal analysis. Spatial join operator Near and the join condition of spatial distance of 6km is used to analyse whether the spatial location of any two airplanes is within 6km, i.e., the spatial association relationships of the global airplanes is determined. Temporal join operator Near and join condition of temporal interval of 10 seconds is used to analyse whether the flight temporal interval of any two airplanes is within 10 seconds, which means the temporal association relationship of the global air flights is determined. The attribute association analysis is to analyse the encounter situation of any airplanes according to their flight number. If interested, attribute statistics of the join result can be computed to obtain the number of early warning flights.

Table 2. Example commands of the spatiotemporal join operation for global daily air traffic movement data

```
./spark-submit --class com.supermap.bdt.main.FeatureJoinMain /home/supermap/com.supermap.bdt.core-9.0.0.jar
--inputTarget {"type":"csv","info":[{"server":"/home/supermap/
/JoinFeatures/flights.csv"}],"specFields":["","time",""],"attributeFilter":"Z>0.0"}
--inputJoin {"type":"csv","info":[{"server":"/home/supermap/
/JoinFeatures/flights.csv"}],"specFields":["","time",""],"attributeFilter":"Z>0.0"}
--joinOperation JoinOneToMany
--joinFields uidx,flight_number,X,Y,Z,time
--spatialRelationship near
--spatialNearDistance 6
--spatialNearDistanceUnit kilometer
--temporalRelationship near
--temporalNearDistance 10
--temporalNearDistanceUnit second
--attributeRelationship target.flight_number!=join.flight_number
--summaryFields [flight_number,Sum]
--output {"type":"udb","server":"/home/supermap/JoinFeatures/result.udb","datasetName":"name"}
```

The spatiotemporal join result for global daily air fight movement data under 6km and 10 seconds join condition is showed on the screen two minutes after submitting the task. Figure 3 illustrates the global air traffic congestion condition within 10 seconds, which can help to optimize air flight management such as adjustment air route schedules. We can also add flight height as attribute join condition, and choose different values of the flight height, which can help to analyse the traffic congestion condition near a particular airport under different heights, since airplanes are commonly ‘parked’ over an airport while waiting for a landing slot.

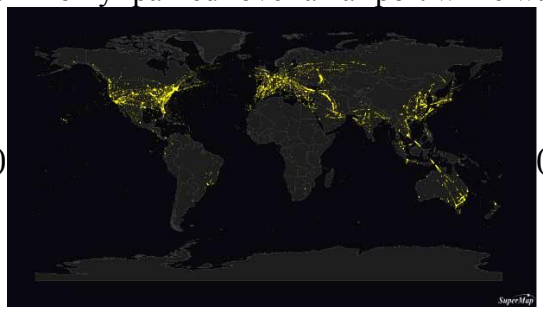
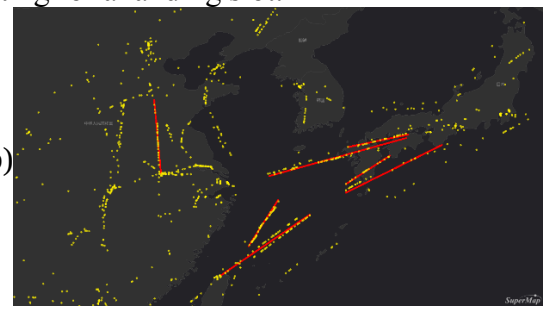
(a)  (b) 

Figure 3. Global air flight congestion of spatiotemporal join analytics: (a) global map, (b) part of the global map

Global flight pattern is discover using the big spatiotemporal analytics engine based on the historical air traffic movement data. In this case, 10 thousand long-distance air routes are extracted from 1 billion traced points based on the temporal information of the points, and the reconstructed trajectory is shown in Figure 4. Global connection analysis is also conducted for Beijing capital airport which is one of the world's biggest air transportation hub (Figure 5).

Figure 4. Trajectory reconstruction for the global air traffic movement data

Figure 5. Global connection analysis for Beijing Capital International Airport

4. Conclusion and Future work

In this study, we designed and implement a big spatiotemporal analytics engine for air traffic movement data. The engine supports 6 spatial join operators, 14 temporal join operators, and attribute join operators. It also supports trajectory reconstruction and a broadcast spatiotemporal analytics operation. Benchmark test shows the high performance of the engine. Our work is important for airport planning and for the determination of future capacity requirements. Future work is to make the engine open source and test it under other movement data scenarios.

Acknowledgements

This work was supported by the National Key R&D Plan (2016YFB0502004), Project of Beijing Excellent Talents (201500002685XG242), National Postdoctoral International Exchange Program (Grant No. 20150081).

References

Airports Council International, 2016, World Airport Traffic Forecasts 2016–2040

Yu J, Wu J, Sarwat M, 2016, A demonstration of GeoSpark: A cluster computing framework for processing big spatial data. *IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 1410-1413.

Tang M, Yu Y, Malluhi Q M, et al, 2016, Locationspark: A distributed in-memory data management system for big spatial data. *Proceedings of the VLDB Endowment.* VLDB, 9(13): 1565-1568.

You S, Zhang J, Gruenwald L, 2015, Spatial join query processing in cloud: Analyzing design choices and performance comparisons. *2015 44th International Conference on Parallel Processing Workshops (ICPPW)*. IEEE, 90-97.

Hagedorn S, Götze P, Sattler K U, 2017, The STARK framework for spatio-temporal data analytics on spark. *Proceedings of the 17th Conference on Database Systems for Business, Technology, and the Web (BTW 2017)*, BTW, 123-142.

Wang S, Zhong Y, Lu H, et al, 2017, Geospatial Big Data Analytics Engine for Spark. *Proceedings of the 6th ACM SIGSPATIAL Workshop on Analytics for Big Geospatial Data.* ACM, 42-45.