Commentary on Polygon Overlay using MapReduce and Random Forest

Prerna Rai
Sikkim Manipal Institute of Technology,Rangpo Sikkim

Prativa Rai
Sikkim Manipal Institute of Technology, Rangpo Sikkim

Bijoy Chhetri
CCCT, Chisopani Sikkim

ABSTRACT
Geographical Information System is known for its analytical capability. The spatial analysis function of GIS is the major factors that distinguish the GIS from other information system. This analysis provides with the functions such as spatial interpolation, buffering, and overlay operations [2]. Among all these functions, the focus on the study here is on overlay operation for polygon on polygon, using MapReduce and Random Forest. The capability to overlay multiple data layers in a vertical fashion is the most required and common technique in geographic data processing. In this study, we focus on the comparison between polygon overlay operation with and without spatial index in MapReduce and also proposed method for polygon overlay using Random Forest and its comparison with MapReduce algorithm.

Keywords
MapReduce, Polygon Overlay, Random Forest, Spatial index

1. INTRODUCTION
GIS overlay operation is one of the major geospatial data analysis methods. Overlay operations can be point on polygon, line on polygon and polygon on polygon. The study here is based on polygon on polygon. Polygon overlay operation is the process of integrating two or more different thematic map layers of same geographic area for GIS analysis [2]. When the process of overlay operation is to be performed for large number of polygons, GIS needs to incorporate Big Data paradigm and one of the programming paradigm that provides a platform for performing overlay operation for polygon in Big Data is MapReduce and Random Forest.

MapReduce has two phases, Map and Reduce phase. The input to the Map phase is the polygon Ids from subject layer and clip layer. An overlay operation over the two layers is carried out and provided with the overlay computation. The computation can be A AND B, A NOT B, AOR B or A XOR B. The MapReduce can compute using Spatial Index or without using Spatial Index. The algorithm comparison for Map Reduce with and without spatial index has been discussed in this paper. It is seen that the efficiency of an operation improves with the use of spatial index compared to the algorithm without spatial index.

The other approach used for computation of polygon overlay is the Random Forest. The use of this special classifier algorithm helps in providing analysis with the uncorrelated data unlike in MapReduce. It is an efficient algorithm that can be executed in parallel environment. It is also efficient in handling imbalanced datasets. The amount of data to be processed is often too large for computation by simple Random Forest. Therefore Random Forest can be implemented using Map Reduce [2][4].

2. POLYGON OVERLAY- AN OVERVIEW
Polygon overlay processing is a common operation for vector data. Polygon overlay operation combines the input polygons from two different map layers named subject one and clip one, respectively of the same geographic area. The output is a third layer represented by output polygons. In general, layers of the same region are stacked together to execute polygon overlay processing.

A. Polygon Overlay processing Algorithm
In [1], the polygon overlay processing algorithm has been discussed. The algorithm is given below:-

Algorithm 1:

1 For all polygons P_A in the subject layer do

2 For all polygons P_B in the clip layer do

3 Compute P_O =overlay (P_A, P_B)

4 Add P_O to Set of output polygons *S*O

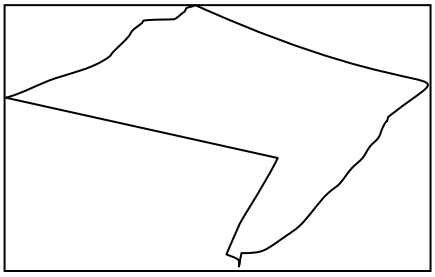

Figure 1: Polygon and its MBR

In addition to the operation, in order to reduce the time of polygon overlay the Minimum Bounding Rectangle (MBR) is used [1]. It is an expression of the maximum extents of a two-dimensional (2-D) object within its2-D (x, y) coordinate system. The MBR of a polygon is shown in Figure. 1.

3. MAPREDUCE ALGORITHMS FOR POLYGON OVERLAY PROCESSING
With the rapid development of GIS, the amount of spatial data is increasing every day [1]. Therefore in order to store and process this data, concept of Big Data can be used. MapReduce is one such Big Data programming paradigm that can be used to process huge volume of data using parallelism. MapReduce Algorithm can be used to carry out polygon overlay processing when there exist huge no of polygons. There are two basic MapReduce Algorithm that can be used for polygon Overlay [1]:

A. MapReduce without using spatial index

B. MapReduce using spatial index.

Spatial index determines the relationship between two geometries that can often be sped up by first determining the relationship between the bounding boxes of the geometries. Indices take this concept one step further by indexing the bounding box of the geometry rather than the geometry itself . Spatial index is represented in the form of grids. The grid index divides the study area by horizontal lines and vertical ones to equal size and unequal grid. Each grid can be viewed as a barrel which records the spatial entities ID of each grid area. [1]

Every grid is indexed by an Identification number. In the Figure 2 below, several polygons have been placed into a 4x4, grid. The cells are numbered from 1 through 16, starting with the upper-left cell.

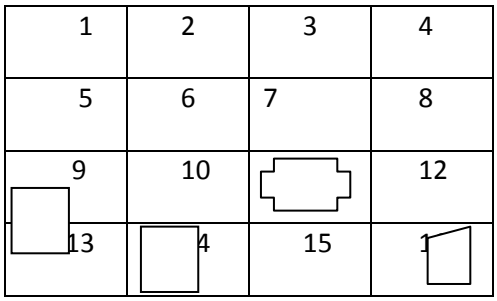

Figure 2.Layer with Grids

A. MapReduce algorithm without using spatial index.
The algorithm is implemented in two distinct phase as shown in Algorithm 2 for two layers, they are Subject layer and clip layer as shown in the Figure 3 and Figure 4 below.

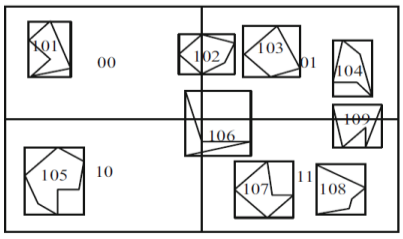

Figure 3. Subject layer

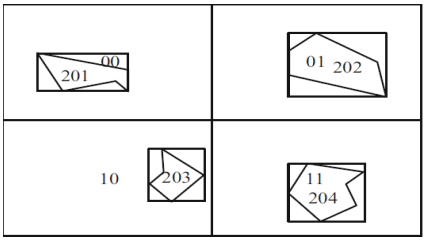

Figure 4. Clip Layer

The MapReduce key value pair is composed of a polygon from clip layer and from subject layer. The two polygons are tested by their MBRs. The polygon ID of clip layer and of subject layer constitutes the intermediate key-value pair which acts as the output of map phase. Then, a merger of the intermediate key-value pairs with the same ID of clip layer polygon is conducted in the reduce phase. Each reduce task executes polygon overlay computation to obtain the intersection of two polygons. Due to no spatial index in this algorithm, a polygon in clip layer must to do overlay computation with all polygons in subject layer. [1]

Algorithm 2. MapReduce without Spatial Index[1]

```
1  Method MAP (Polygon_C_ID, Polygon_S_ID)
2  {
3     if current two polygons' MBRs intersect
4        EMIT: (Polygon_C_ID, Polygon_S_ID)
5  }
6  Method REDUCE (Polygon_C_ID, list(Polygon_S_ID))
7  {
8     for each polygon P_S in the list
9        compute P_O =overlay(P_S, P_C)
10       if P_O does exist
11          add P_O into S_O
12 }
```

B. MapReduce algorithm using spatial index.
The MapReduce algorithm using spatial index performs polygon overlay computation in two distinct phases.

i. First phase: Build spatial grid index

ii. Second phase: Polygon overlay computation.

i. First Phase: Building Spatial grid index

During this phase spatial grid index is build by using grid table as shown in Table 1 and layer table in Table 2. for both subject and clip layer.

The input to Map phase is the set of key-value pair, containing a <polygon's ID, data> which produces an intermediate key value pair consisting of <grid's ID, polygon's ID>. This intermediate key-value pairs are accepted by the Reduce task. As a result, each reduce task obtains the key-value pair consisting of a grid's ID and the list of all polygons in the grid.

Table 1. Grid Table

Grid ID	List of Polygon ID
00	101,102,106
01	102,103,104,106,109
10	105,106
11	106,107,108,109

Table 2. Layer Table

Polygon ID	List of Grid ID
101	00
102	00,01
102	01
104	01
105	10
106	00,01,10,11

107	11
108	11
109	01,11

ii. Second Phase: Polygon overlay computation.

During this phase polygon overlay computation is carried in two distinct phases of Map and Reduce as shown in the Algorithm 4 below.[1]

Algorithm 4: Pseudo code of MapReduce procedure for overlay computation[1]

```
1 Method MAP (Grid_ID, ListS (Polygon_ID))
2 {
3       for each one in ListC of this grid
4       for each one in ListS of this grid
5       if current two polygons' MBRs intersect
6       EMIT: (PolygonC_ID, PolygonS_ID)
7}
8 Method REDUCE (PolygonC_ID, list (PolygonS_ID))
9 {
10 for each polygon PS in the list
11 compute PO =overlay (PS, PC)
12 if PO does exist
13 add PO into SO
14}
```

From the above algorithm it is seen that the input to the Map phase is the key-value pair with each grid's ID and the list of subject layer polygons in the grid. It is pair wise-tested to check whether they are overlapped or not based on their MBRs and output the intermediate key-value pairs consisting of <clip polygon's ID, subject polygon ID> which goes as an input to Reduce task.

The Reduce phase than obtains the key-value pair containing the resultant <clip polygon's ID, List of ID of all subject polygons> in this grid.

4. ALGORITHM ANALYSIS

In reality there exists many map layers where the number of polygon is huge and performing overlay computation between layers is time consuming. Therefore handling huge amount of data introduces the role of big data. The algorithm of MapReduce without spatial grid index do not divide the layers into equal size but the MapReduce algorithm with spatial index uses grid to partition one layer into squares of equal size. Hence, the overlay computation is carried out between each polygon in a square of subject layer and that in the same square of the clip layer. In this way, the times of calling intersection computation of the GPC is significantly reduced than in the Map Reduce without spatial index. [1]

A. Comparison between MapReduce with and without spatial data.

As per the Experimental study in [1] Comparison is made between the algorithm of map reduce using spatial data and without using spatial data, to verify the efficiency of spatial grid index as depicted in Table 3. The difference between the two is based on running time in single node and running time on different nodes. 12 Nodes are used in the Experiment. No of polygon in subject layer and clip layer is M and N. Therefore the times of calling intersection computation in the map phase of the first algorithm is M × N and computation in the reduce phase is N. Furthermore, the times of calling intersection computation in the map phase of the second algorithm is the number of grids and that in the reduce phase of second algorithm is N.

It is also noted that the average running time of MapReduce with spatial index is comparatively lesser than without spatial index.

Since spatial index is efficient in searching process, it is also possible to handle huge amount of data for execution compared to the computation without spatial index. Therefore for the efficient and faster polygon computation MapReduce with spatial index outperforms the MapReduce without spatial index. The overhead incurred is reduced to greater extent.

Table 3.Comparison between 'With' and 'Without' grid Map Reduce

Parameter	With Spatial Index	Without Spatial Index
Times of calling intersection computation in map phase[1]	1048576 (NO OF GRIDS)	14259443380(MXN)
Times of calling intersection computation in reduce phase[1]	117439 (N)	117439 (N)
Average of running time(S)[1]	166	1537
File format	HBase	.shp to Text input
Data size	Large (1.23 GB)	Medium (800 MB)

5. RANDOM FOREST

The other approach used in Big Data programming paradigm for polygon overlay process is Random forest. It is a collection of many un-correlated decision trees. It acts as a classifier and operates by constructing many decision trees at training time and outputting the class that is the mode of the classes of the individual trees. Each tree gives a classification, and each tree "votes" for that class. The forest chooses the classification having the most votes over all the trees in the forest[11].

This algorithm works with random and de-correlated data's. It makes use of decision tree which determines the node of the tree based on the information gain and entropy. Each tree is grown as follows [10]:

A. Generating each tree and Forest:

- Given a dataset N, and M input variables, select m from sample S dataset.
- m is the number of input variables and m is less than M.
- Select m variable at random '*with replacement*' from M for S1.
- Select the best split on these m.
- The value of m is constant during the forest growing.
- Each tree is grown to the largest extent possible.
- Averaging all the trees to form a forest.

B.Features of Random Forests[10]:

- It runs efficiently on large data base and can handle thousands of input variables.
- It gives estimates of what variables are important in the classification.
- Generated forests can be saved for future use on other data.
- The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection.
- It offers an experimental method for detecting variable interactions.

Random forest is found to be fast in execution. Running on a data set with 50,000 cases and 100 variables, it produced 100 trees in 11 minutes on a 800Mhz machine [10].

6. PROPOSED RANDOM FOREST APPLICATION IN POLYGON OVERLAY

Random forest can be also be used for computing polygon overlay process and find its role in GIS. The process for building a forest from a tree is as shown in the Figure below.

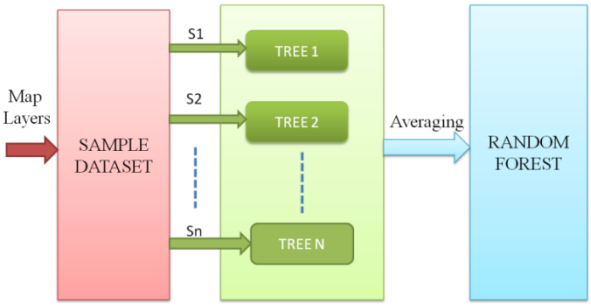

Figure 5. Block Diagram of Random Forest generation

A. Building a Tree:

Considering:

i. Subject and Clip layer

ii Polygon with M number of attributes and N no of datasets.

do

select sample dataset S1

select m attribute from M where m<=M in S1i.e {A1, A4,A5}

choose the best among these tree attributes... say A4.

choose A4 as root node.

select a random split value.

split S1 on root node A4.

select sample s11 and s12 as new sample data set.

select m for S11 as {A2, A5, A6}

choose the attribute.. say A5.

select m for S12 as {A1,A2,A7}

choose the attribute ... say A2.

Repeat the process until all leaf nodes are exhausted.

END

The tree is build as shown in the Figure 6 below. Similarly the tree for clip layer can also be designed. At the end of the run jth attribute is taken to be the class that got most of the votes from the forest.

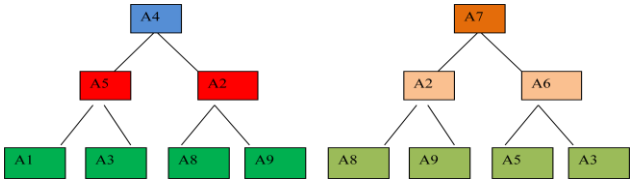

Figure 6. Decision Trees

7. COMPARISION BETWEEN RANDOM FOREST AND MAP REDUCE FOR POLYGON OVERLAY

In Table 4, Random Forest have many property which MapReduce do not and MapReduce have many features which Random Forest do not.

Table 4. Comparision between Polygon overlay using MapReduce and Random Forest.

METRICS	RANDOM FOREST	MAPREDUCE	REMARKS
Parallel environment	RF can be executed in parallel environment	MR can be executed in parallel execution	According to the study in [11]
Scalability	Not fit for massive data	Effective framework for processing large datasets in parallel environment	As specified in [12]
Correlation	Works well with un correlated data	Works well with correlated data.	As described[14]
Imbalanced Datasets	Good with imbalanced data set	Not good with imbalanced datasets	Deduced in[13]

Therefore when Random forest is implemented using Map Reduce it can increase the computation capability of the polygon overlay.

8. CONCLUSION

The usage of distributed system in the field of Geographic Information System (GIS) is the increasing need of hour due

to the rapid increase in the volumn of spatial data. To analyse and perform operation on this large data is time consuming using the desktop application. Therefore in order to perform Polygon Overlay operation, which is one of the function of GIS and to meet up with the challenges of growing data, the concept of Map Reduce is used here. This concept works on the processing of task in parallel. The Map Reduce perform overlay of two polygon with and without using spatial grid index. It is found that the use of grid index and parallel task execution by Map Reduce has significantly increased the performance of the overlay process and reduced the time for execution considerably.The polygon overlay computation can also be performed using a special algorithm called Random Forest. Random forest is based on its special feature "Randomness" in selecting the data and split value. The algorithm can be used with Map Reduce to increase the scalability property of Random forest.

9. REFERENCES

[1] Yong Wang · Zhenling Liu · Hongyan Liao ·Chengjun Li, Improving the performance of GIS polygon overlay computation with MapReduce for spatial big data processing, Springer Science+Business , Accepted: 10 January 2015Media New York 2015, DOI 10.1007/s10586-015-0428-x

[2] Basudeb Bhatta,"Remote Sensing and GIS", Second Edition in 2011, ISBN-10:0-19-807239, 554-558.

[3] Puri, Satish, "Efficient Parallel and Distributed Algorithms for GIS Polygon Overlay Processing." Dissertation, Georgia State University, 2015. http://scholarworks.gsu.edu/cs_diss/98 accepted for inclusion in Computer Science Dissertations by an authorized administrator of ScholarWorks @ Georgia State University.

[4] Puri, S., Agarwal, D., He, X., Prasad, S.K.: MapReduce algorithms for GIS polygon overlay processing. In: Proceedings of the 27th IEEE International Parallel and Distributed Processing Symposium Workshops and PhD Forum (IPDPSW), pp. 1009–1016 (2013)

[5] Guo , B. Palanisamy, H. A. Karimi ,A Distributed Polygon Retrieval Algorithm using MapReduce, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume II-4/W2, 2015 International Workshop on Spatiotemporal Computing, 13–15 July 2015, Fairfax, Virginia, USA

[6] Dinesh Agarwal, Satish Puri, Xi He, and Sushil K. Prasad1, A system for GIS polygonal overlay computationon Linux Cluster - An experience and performance Report, 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum

[7] Fangju Wang, A Parallel Intersection Algorithm for Vector Polygon Overlay, 0272- 17.16.93/0300-0074 IEEE Computer Graphics & Applications

[8] Cheng, C.: Spatial Database Management System. Science Press, Beijing (2012)

[9] Processing of massive data, MapReduce New Trends In Distributed Systems MSc Software and Systems

[10] Random Forests Leo Breiman and Adele Cutler. www.stat. berkeley. edu/breiman/ RandomForests date: 29.04.16

[11] Random forests and big data Robin Genuer, Jean-Michel Poggi, Christine Tuleau-Malot. Random forests and big data. 47`eme Journ´ees de Statistique de la SFdS, Jun 2015, Lille, France. 2015. HAL Id: hal-01160643 https://hal.archives-ouvertes.fr/hal-01160643 Submitted on 8 Jun 2015

[12] A Scalable Random Forest Algorithm Based on MapReduce Jiawei Hanl,Yanheng Liul,College of Computer Science and Technology, Jilin University 2,Changchun University,Jilin Province, ChinaJason.hjw@gmail.com978-1-4673-5000-6/13.2013 IEEE

[13] On the use of MapReduce for imbalanced big data using Random Forest Sara del Río, Victoria López, José Manuel Benítez, Francisco Herrera Dept. of Computer Science and Artificial Intelligence, CITIC-UGR, University of Granada, Granada, Spain. Article history: Received 28 March 2013Received in revised form 21 February 2014 Accepted 11 March 2014 by Elsevier.

[14] Random forests, aka decision forests, and ensemble methods Published on Feb 21, 2013