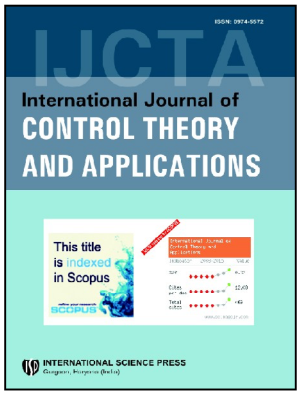


International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Special Issue, 2016

BigGSDA: Big Geo Spatial Data Analyser

[1]K. Kishore Raju, [1]K. Swamy, [2]D. Rajyalakshmi and [1]G. P. Saradhi Varma

[1] *SRKR Engineering College, Bhimavaram, India*
[2] *JNTU, Kakinada*
E-mails: kkrsrkrit@gmail.com; kswamyit@gmail.com; jrajyalakshmi@gmail.com; gpsvarma@gmail.com

Abstract: From the ancient years' data, distributing process has reached many areas of computer science with geographic information system. With the enduring increase of existing data, algorithms and data management are to be moved to a new design, which requires a great deal of effort. In the existing AEGIS Frame work, large image data is distributed over the different hosts. Distributing the data over different machines and parallel computing will increase the execution time. This paper defines a geospatial data processing framework Big Geo Spatial Data Analyzer (BGSDA) to enable distribution of raster data in HDFS environment and pixel based classification using MapReduce environment. Multi Spectral Geo spatial satellite images are split into equal and individual blocks using edge detection based border handling strategy which uses 3x3 filtering kernel and they are stored in a Hadoop cluster as part of data distribution. On each image split, parallel processing is implemented by MapReduce to process pixels. Based on Known sample pixels data each unknown pixel in each image split will be classified using Map(s) and Reducer(s) concurrently. After classification of each image split in the individual data node, all are merged by matching the borders of each split using edge detection algorithm to get a single large classified image. It was observed that processing time has been reduced drastically in the proposed system when the satellite images are too large and Number of Data Nodes are increased.

Keywords: Multi Spectral Image, Splitting, Merging, Pixel classification, Edge detection, Parallel processing

1. INTRODUCTION

Big data is a well-known term discussing about data sets becoming extremely very huge and complex. For handling this type of data needs high performance computing or distributed data processing. Nowadays, though we are having several concepts have been developed, most industries are using Hadoop Map-Reduce model as it is open-source implementation and able to handle the large data sets also.

In terms of geographical information system image analysis, new concepts have been developed and applied successfully in multiple cases. Earlier solutions have been developed using small data sets and on single machine execution in mind, which is not applicable to big data.

In Map-Reduce environment huge satellite image was taken as an input. And the image is split into equal number of tiles. The split image data are distributed to the HDFS for parallel computation which will be efficient and run successfully. HDFS use Map-Reduce for parallel and distributing data processing method for performance on a computer group. It involves two functions, map and reduce. The key/value sets are the inputs to Map function. When a Map-Reduce assigns a job to the system, the map gets started on the compute nodes, for each map task applies the map function to every key/value pair that is allotted to it. Zero or more intermediary key/value sets can be generated from the same input key/value pair. Intermediary results get stored in a local file system and arranged by keys.

After completion of all the map tasks, the Map-Reduce advises the reduce tasks to start processing it. The reducer yields the output files from the map in-parallel, and merges the files achieved from the map to combine the key/value sets into a set of new key/value pair, where all values with the same key are clustered into a list and used as input to the reducer function. The reduce job applies the user-defined logic to process the data. The results, usually a list of values, are written back to the storage system.

From the historical review cluster computing has become very broadly popular, in which data-parallel computations are executed on clusters of untrustworthy machines by systems that spontaneously provide locality aware setting up, fault tolerance and load balancing [1]. Big data refer to the vast amount of data composed over time that are very difficult to examine and handle by using common DBMS tools [2]. Big data can yield enormously useful information, it also presents new encounters with esteem to how much will cost the data to be secure and maintained for long perseverance [3]. To process the huge amount of data for managing and processing of distributed data there are some tools/frameworks. Hadoop is an unrestricted, open source Java programming framework that supports the processing of large data sets in a scattered computing environment, sponsored by the Apache Software Foundation [4]. Geospatial information has extended many zones of science and found many application areas while the focus was moved to distributed and high concert computing [5]. The distributed geographical information system has many advantages; current open-source mechanisms can be re-claimed in a distributed environment. Hadoop offers a distributed file system and a framework for the analysis and makeover of very massive data sets using Map-Reduce [6]. Map-Reduce model has been successfully applied to spatial indexing [7], for pull out Geo–locations from GPS (Global Positioning System) data. Spatial Hadoop [8] spreads functionality with well-organized spatial data storage, indexing and spatial query support. Hadoop GIS proposals complete spatial data warehouse explanation over HIVE [9] (Data warehouse Infrastructure). By using raster data in image processing also been magnificently applied to Map-Reduce model, by using a pixel based provision of images, the Map task is used for both as a locator and initial processor while Reduce job performs result summary. Powell [10] defines how the NASA grips image processing of celestial images took by Mars Orbiter and rovers. Concise reports are provided about the division of Giga Pixel images into tiles, based on his review the image is split into tiles and processed. Wang et al [11] discuss how to speed up the analysis of material microarray images by replacing human expert analysis of computerized processing algorithms. Most of the algorithms had been adapted for a specific tool. This lead to the concept of a toolkit, i.e. Hadoop environment, which works on distributed data, in addition to that, this tool also offers proper data management infrastructure which enables the storage and retrieval of distributed geospatial data. This tool has been introduced [12], and developed as an extension to the AEGIS framework. For splitting and merging of an image Valencia et al [15] proposed an Edge detection algorithm for data partitioning of a multi spectral image in spatial domain using filters with 3 x 3 kernel size. For merging the image same edge detection algorithm is used in order to form an actual image without any loss of pixel data at the merging edges.

2. PROPOSED SYSTEM

In this paper LISS-IV satellite image was taken as an input. Satellite images are very huge in size (in terms of Memory) so the image is split into the number of tiles based on size. After splitting of image into equal number of tiles, these tiles are distributed to HDFS (Hadoop Distributed File System), thereby it starts execution with the given input specifications and limitations. The input contains location coordinates and pixel information, etc.

those are available in the metadata catalogue. Built on the contribution limitations, the procedure is selected from the metadata. In Hadoop Map-Reduce execution, the Mapper reads the input file from metadata catalogue. These files all are together arranged the input of mappers to the nodes where the data are positioned. The mapper is a program which performs the operation on the same node; all input files are processed synchronously by mappers. Mapper results are accelerated to the reducers for supplementary processing. Reducer remains also a program which completes the operations nearby on the name node. And the data are sent to the output writer. Lastly, the output is created in the pixel color that classifies the image For Example; blue color belongs to class-A (water) etc. And finally merge the image as same as original image taken as input.

Data Format: Data storing format is stretchy along with divider strategy. All types of format can be quantified, i.e. the chattels of the input image (Geometry coordinates, RGB values, Image magnitudes) can be hoarded into the metadata catalogue along with the pleased (vector and raster data) that can be saved in a binary format. Almost all the organizations are using a standard format to store the spatial data in their specific presentation, i.e. Relational Database, and further statistics can also be kept in metadata catalogue.

Datasets can be a single large file or multiple small files. The original image will be divided into small tiles. Information can be circulated by using a spatial division method, such as KD-tree or quad tree. While splitting the image into n tiles there may be a chance of losing the pixels. So that's better to maintain the duplicates, these reproductions can be detached, while measurements about the foundation, the information is preserved in the metadata catalogue. This methodology can also be applied for any multifaceted data illustration. By maintaining edge vertices duplicates and the basis of the data, the native structure can be restructured, in case if an edge cut. Metadata of the segregated geometries includes the figures about the beginning and partition approach and stored in the central metadata catalogue.

In a scattered environment, multiple Map-Reduce developments are executed as a single process in Hadoop. For the duration of execution the workflow generates several intermediate results, and these results are repeatedly shambled between the progressions by Hadoop. HDFS main goal or designed for write once and read many times.

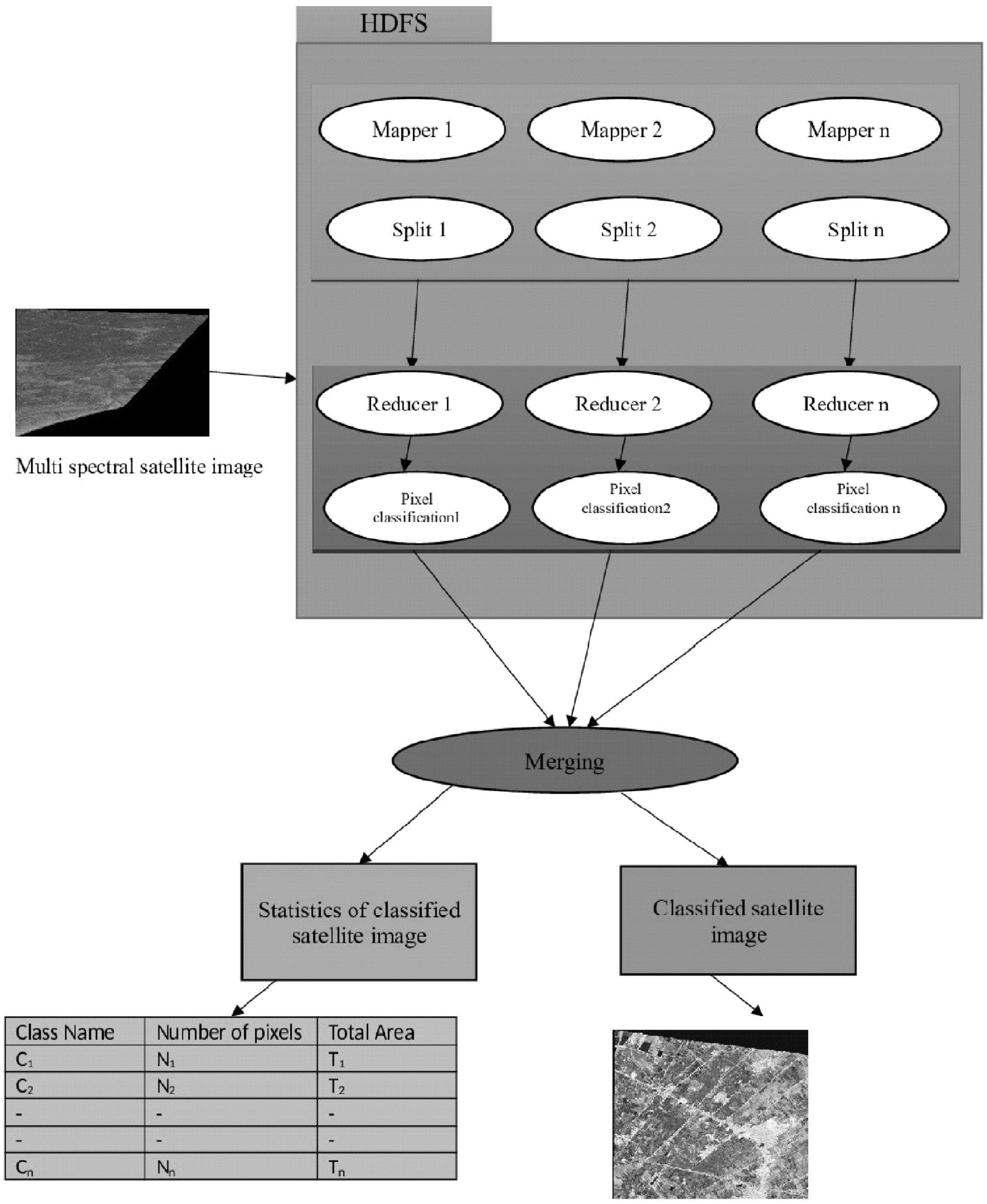
Big Geo Spatial Data Analyzer is implemented in the following modules

- Splitting satellite image into specified number of parts using Indexing.
- Data distribution by loading the image splits in HDFS in Hadoop cluster.
- Parallel Processing using Map Reduce for Pixel classification on a Hadoop cluster.
- Merging

Map

Here in this context map takes image as an input. For the image it takes key, value and context as an argument to the map method. Finally, it splits the original image into small tiles as an output.

```
public void map(LongWritable key, BufferedImage value, Context context) {
net.semanticmetadata.lire.imageanalysis.filters.CannyEdgeDetector ced = new net.semanticmetadata.lire.imageanalysis.filters.CannyEdgeDetector(in, 40, 80);
ImageIO.write(ced.filter(), “jpg”, new File(“hadoopedges/out”+v+”.jpg”));
/* it splits the image into small tiles and saves the each and every split image into the given path. */
Path newimgpath = new Path(context.getWorkingDirectory(), context.getJobID().toString()+”/”+key.get());
dfs.createNewFile(newimgpath);
FSDataOutputStream ofs = fs.create(newimgpath);
ImageIO.write(edges, “JPG”, ofs);
/* it saves the image as JPG into the provided path.*/ }
```



Class Name	Number of pixels	Total Area
C_1	N_1	T_1
C_2	N_2	T_2
-	-	-
-	-	-
C_n	N_n	T_n

Figure 1: Architecture diagram for Big Geo Spatial Data Analyzer (BGSDA)

(A) Splitting image using edge detection based border handling strategy

To distribute the multi spectral image in spatial domain needs data partitioning mechanism and indexing. To partition the data, edge detection based border handling strategy is implemented to avoid the forfeiture of data between the neighbour splits. For this 3 x 3 filtering kernel is used for duplication of pixels at the processors to avoid border possessions. The no of replicated pixels R_p can be computed by using the below equation 15.

$$Rp = 2 * \left[\left(2 \left[\frac{\log 2\ P}{2} \right] t \right) - 1 \right] * Nr + 2 * \left[\left(2 \left[\frac{\log 2\ P}{2} \right] \right) - 1 \right] * Nc \qquad (1)$$

Where P = Number of partitions

Nr = Number of rows in actual image.

Nc = Number of columns in actual image.

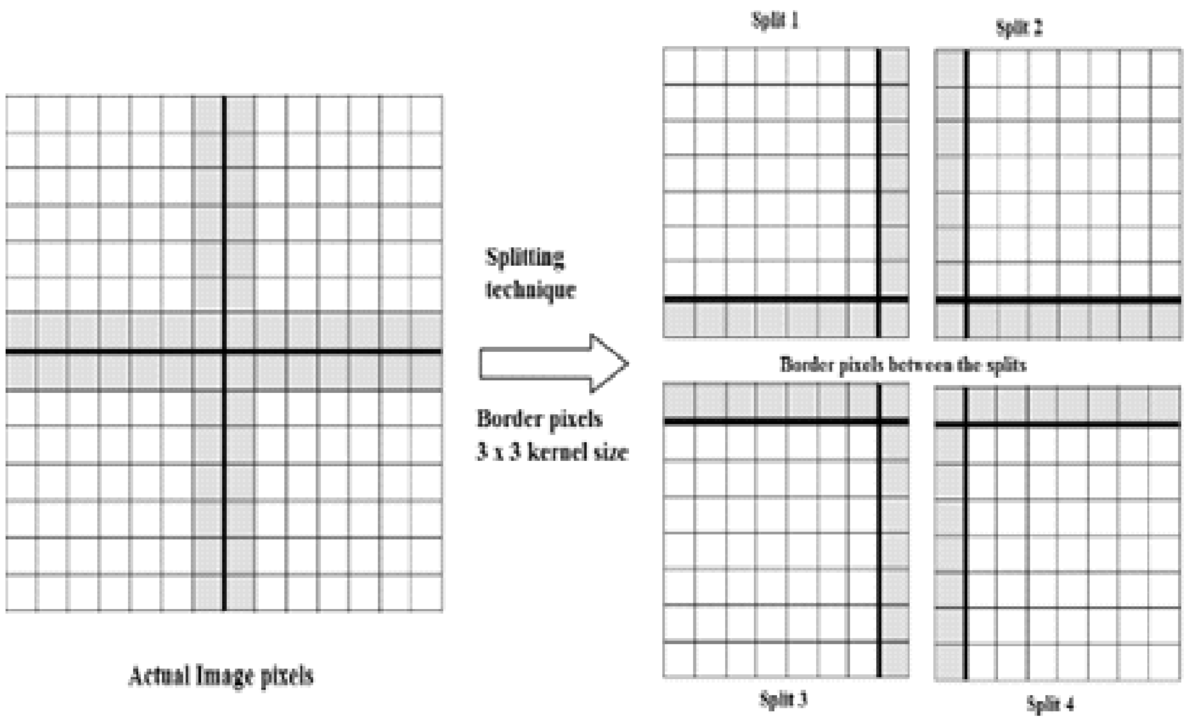


Figure 2: Edge detection based border handling strategy

To provide the spatial component of the split images R* tree indexing mechanism is implemented. Based on R* Tree indexing splitting of an image and indexing was done.

Algorithm for *searching*

Input: Image Coordinates and Dimensions.

Output: Return a list of objects whose dimensions overlap with the given coordinates.

```
RangeSearch(TypeNode RN, TypeRegion Q1)
Step 1: if RootNode not equals to a leaf node then
Step 2: observe each item of RootNode
Step 3: for each such entry e call RangeSearch(e1.ptr1,Q1)
Step 4: else
Step 5: scrutinize all entries and find those for which e.mbrintersects Q1
Step 6: add entries to set A1
Step 7: Endif
```

Algorithm for *deletion*

Input: Image Coordinates, Dimensions and entry node to delete.

Output: Returns true if node deletes from tree else returns false.

```
Delete(TypeEntryE, TypeNodeRN)
Step 1: if RootNode equal to leaf node then
Step 2: Find every entry of RootNode to know E.
Step 3: else
Step 4: Search all entries of rootnode which cover E.
Step 5: delete E from Leaf
Step 6: endif
Step 7: Condense_Tree(L)
Step 8: if root node have one child then
Step 9: delete root node
Step 10:endif
```

Algorithm for condense Tree

Input: Node to be inserted

Output: Reconstructs tree.

```
Condense_Tree(TypeNodeL1)
Step 1: Set X1 = L1
Step 2: whileX1 not equal to root node
Step 3: parentX will be the root node to X1
Step 4: ifX1 < m1 entries then
Step 5: delete EX
Step 6: add X1 to N
Step 7: endif
Step 8: ifX1 not deleted then
Step 9: reconstruct tree
Step 10: endif
Step 11: Set X1 as parentX
Step 12: endwhile
```

Node entry deletion from an R-tree is done by above algorithm. There is a difference between B+ tree and R – tree in under flow condition. Even if one need to merge the two R-tree nodes then re-insertion is happened due to following reasons.

- Reinsertion and merging gives the same output. Moreover, the algorithm also be used for insertion. Accessing of disk information in deletion operation is most important as it effects the performance, so buffer memory should always be available for re-inserting the elements.
- The Insert algorithm always try to maintain quality of tree. Consequently, if re-insertion takes place also quality of tree will not decrease.

(B) Pixel Based Classification using Maximum Likelihood Classifier (MLC)

Pixel based Classification [14] is done to mine the patterns like water body, agriculture land, trees, living area, roads, Bare lands etc. Almost similar behaviour pixels are grouped as one class. More precisely, image objects are groups of pixels that are almost similar to spectral chattels (i.e., colour), size, shape, and texture, as well as context from a neighbourhood surrounding the pixels. Maximum Likelihood Classifier (MLC) [13] also called as Bayesian classifier. Where pixels are classified only of the group of pixels belongs to same like hood.

$$L_k = \frac{p(k1) * p\left(\frac{k}{x1}\right)}{p(i1) * p\left(\frac{k}{i_1}\right)} \tag{2}$$

Where, $p(k1)$ = Prior probability

$p\left(\frac{k1}{x1}\right)$ = Restrictive probability function of class K.

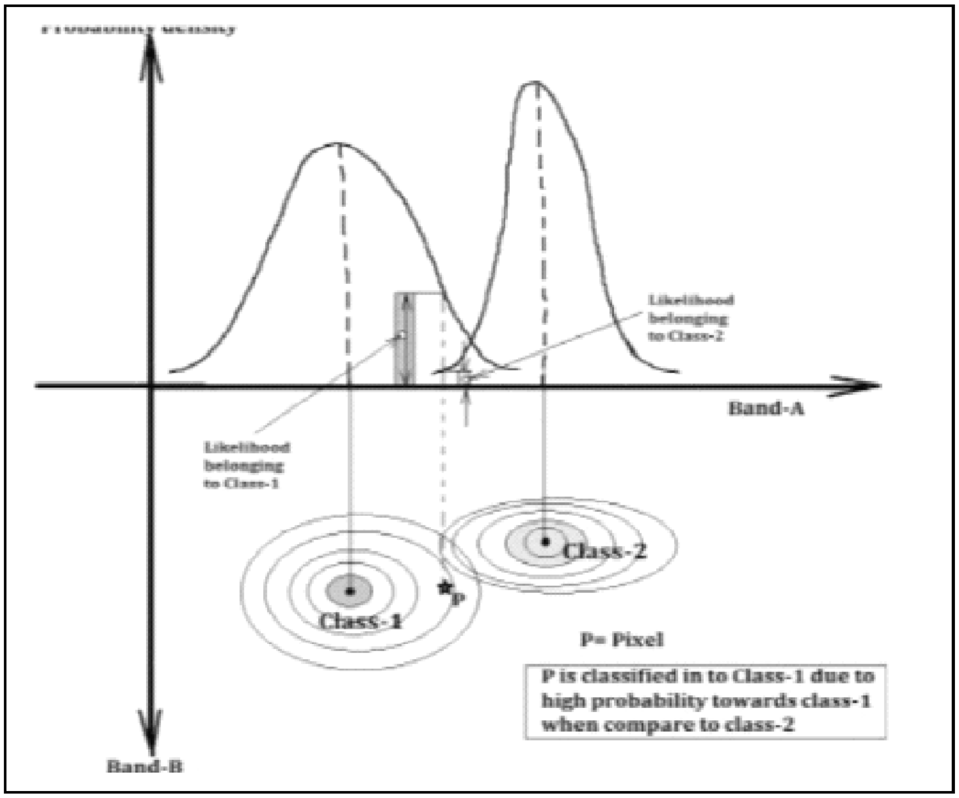

Probability density
Likelihood belonging to Class-2
Band-A
Likelihood belonging to Class-1
Class-2
Class-1
P
P= Pixel
P is classified in to Class-1 due to high probability towards class-1 when compare to class-2
Band-B

Figure 3: Concept of MLC classification

Reduce: After completion of classifying the image, each classified split need to be merged to form actual image. For that, Edge detection based filter of 3 x 3 kernel size algorithm which is used earlier in splitting the image again used to merge the splits. Merging is done by comparing the edge pixels of neighbour splits, if they are matched will be merged without any loss of pixel data.

C. Algorithm for *Merging*

```
Input: key and iterator values.
Output: Image.
Step 1: reduce (key, Iterator values):
Step 2: selects the hdfs path
Step 3: repeat loop until it finish all values
Step 4: get the tile bytes
Step 5: set RGB values
Step 6: close();
Step 7: collects the output to data output stream
Step 8: write the final output to path
Step 9: endm
```

3. EXPERIMENTAL RESULTS

The proposed Hadoop framework is tested on a cluster of four nodes with replication factor 3. Each data node is configured with core-i3 2Gb RAM and name node is configured with core-i5 4Gb RAM, total four commodity systems. A three bands (B2, B3, B4) multispectral spatial satellite image (LISS-IV) data with 5.8 M spatial resolution. To exhibit the ability of proposed framework, a study area from LISS-IV satellite image of approximately covers 50 x 40 Km^{2}rectangle is located in and around kolleru, West Godavari Dist., AP, India is selected. The study area contains innumerable land shelters like Living area, Water bodies, Agriculture, Bare land, etc. Study Area is positioned between 16°45’42.2"N 81°00’09.5"E and 16°19’51.5"N 81°36’01.3"E (Figure 4).

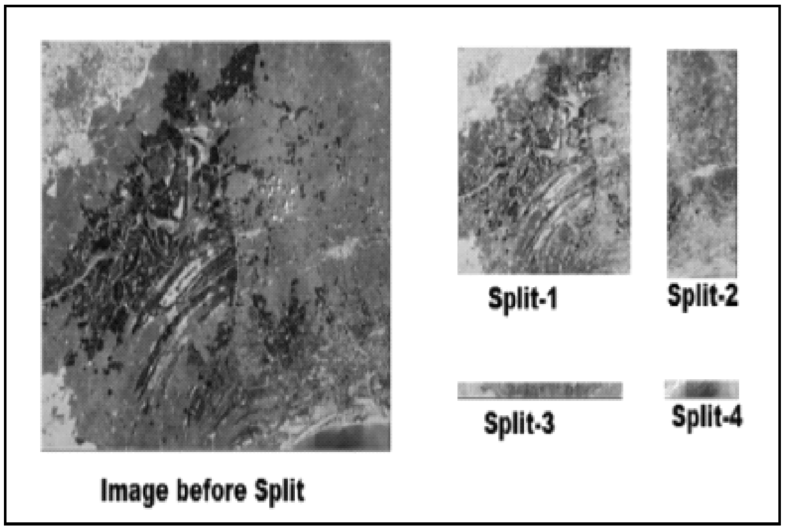


Figure 4: A Three band LISS-IV image divide into four parts and are distributed in different data node

From the selected study area different LISS_IV images with different sizes are selected and tested on existing system and Big Geo Spatial Data Analyser (BGSDA) framework. It is evident that (as shown in Figure), when the input image size increases execution time of existing system increases proportionally. In proposed framework analysis time is less when the size of image increases. In case of small images, the proposed system will perform like traditional system some time even worse.

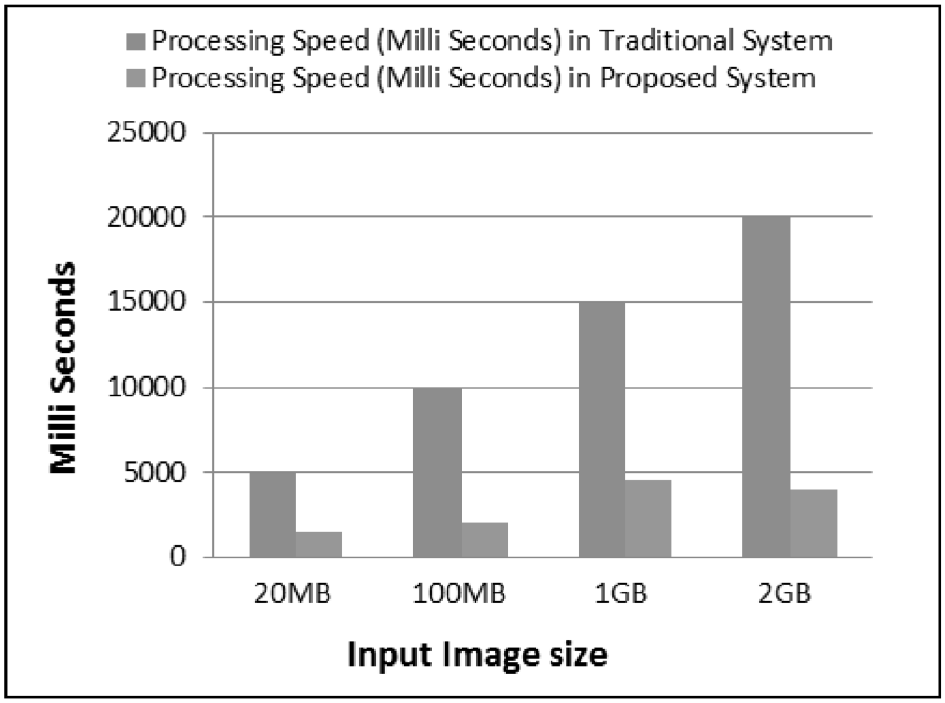


Figure 5: Performance evaluation between existing and proposed systems at different image size

From the above figure we can say that proposed system has better performance when compared to the existing system. For example, if we consider 20MB and 100MB satellite image sizes difference between these two in performance is very high. The existing system has taken 5000MilliSeconds for execution, whereas proposed system has taken only 1500MilliSeconds. Performance wise there is a drastic change in performance. As the size increases, i.e. if the pixel resolution increased them in performance it has vast change. Because in existing system we have only one high configured system so that the single commodity system have to process the huge satellite image which takes more time for execution. Where as in proposed system we distribute the data for the different commodity system all are executed in parallel this way it drastically reduce the execution time when compared to existing systems.

4. CONCLUSION

Classification of very huge multi spectral land cover satellite images (LISS-IV), Big Geo Spatial Data Analyzer (BGSDA) framework is one of the alternatives for effective and efficient mining of patterns like Water bodies, Agriculture, Trees, Living area, Roads and Streams, etc. When the satellite images are too large, BGSDA will perform faster classification (i.e.60 % to 70% reduction in processing time) by creating more number of Mappers and Reducers. It is evident that BGSD analyzer handles very huge raster data easily which is not possible by traditional analyzers. Further, classification performance can be improved by optimizing the Mappers and Reducers.

REFERENCES

[1] MateiZaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, “Spark: Cluster Computing with Working Sets” in Ion Stoica University of California, Berkeley.

[2] SeyyedMojtaba Banaei1, Hossein Kardan Moghaddam2 “Hadoop and Its Role in Modern Image Processing” in Open Journal of Marine Science, 2014, 4, 239-245.

[3] Michael, K. and Miller, K.W. (2013) Big Data: New Opportunities and New Challenges. Journal of IEEE Computer Society, 46, 22-24.

[4] Hadoop. *http://hadoop.apache.org/*

[5] LeeC, GassterS, PlazaA, ChangCI,Huang B. "Recent developments in high performance computing for remote sensing: are view:, IEEEJSelTop Appl Earth ObservRemoteSens2011;4(3):508–27.

[6] HDFS. *http://hadoop.apache.org/hdfs/*

[7] Cary A, Sun Z, Hristidis V, Rishe N. Experiences on processing spatial data with Map Reduce. In: Proceedings of the 21st international conferenceon scientific and statistical database management (SSDBM). Berlin, Heidelberg: Springer- Verlag. ISBN978-3-642-02278-4; 2009. p. 302–19.

[8] Eldawy A, Mokbel MF. Ademonstration of Spatial Hadoop: an efficient mapreduce framework for spatialdata. ProcVLDBEndow2013;6(12):1230–3.

[9] Konstantin Shvachko, HairongKuang, Sanjay Radia, Robert Chansler Yahoo! "The Hadoop Distributed File System" in Sunnyvale, California USA {Shv, Hairong, SRadia, Chansler}@Yahoo-Inc.com

[10] Sridhar Vemula Computer Science Department "Hadoop Image Processing Framework" in Oklahoma State University Stillwater, Oklahoma.

[11] Mark W. Powell, Ryan A. Rossi† and Khawaja Shams "A Scalable Image Processing Framework for Gigapixel Mars and Other Celestial Body Images" in Jet Propulsion Laboratory, California Institute of Technology, {Mark.W.Powell, Ryan.A.Rossi, Khawaja.S.Shams}@ jpl.nasa.gov

[12] Giachetta R. Advancing age ospatial frame work to the Map Reduce model", In: indenbergh R, Spagnuologo M, Boehm J, editors. Proceedings of the 1stI Qmulus workshop on processing large geospatial data; 2014.p.45–52.

[13] D. Rajyalakshmi; K. Kishore Raju; G. P. Saradhi Varma "Taxonomy of Satellite Image and Validation Using Statistical Inference", 2016 IEEE 6th International Conference on Advanced Computing (IACC), Year: 2016, Pages: 352 -361, DOI: 10.1109/IACC.2016.72, IEEE Conference Publications.

[14] M. Krishna Satya Varma; N. K. K. Rao; K. K. Raju; G. P. S. Varma "Pixel-Based Classification Using Support Vector Machine Classifier", 2016 IEEE 6th International Conference on Advanced Computing (IACC), Year: 2016, Pages: 51 - 55, DOI: 10.1109/IACC.2016.20,IEEE Conference Publications.

[15] Valencia, D., P. Mart´inez, A. Plaza, and J. Plaza (2008), Parallel Wild land Fire Monitoring and Tracking Using Remotely Sensed Data, in High Performance Computing In Remote Sensing, edited, pp. 151-182, Chapman & Hall/CRC, Boca Raton, Fla.; London.