# Big Data Frameworks for Efficient Range Queries to Extract Interested Rectangular Sub Regions

Süleyman Eken
Kocaeli University
Computer Engineering
İzmit, 41380, Turkey

Ahmet Sayar
Kocaeli University
Computer Engineering
İzmit, 41380, Turkey

## ABSTRACT
A satellite object can consist of more than one mosaic image. To extract any object from remote sensing satellite images, mosaic images need to be stitched. It is critical problem that which mosaics will be selected for image stitching among big mosaic dataset. In this paper, we propose two approaches to overcome mosaic selection problem by means of finding rectangular sub regions intersecting with range query. Former one is based on hybrid of Apache Hadoop and HBase and latter one is based on Apache Lucene. Their effectiveness has been compared in terms of response time under varying number of mosaics.

## General Terms
Remote Sensing, Distributed Systems, Computational Geometry

## Keywords
Image stitching, Range query, Apache Hadoop, HBase, Lucene, LandSat-8

## 1. INTRODUCTION
Object extraction from remote sensing images has been focused research area for thematic cartography, disaster management, urban planning, by many scientists. While some extracted objects are man-made such as roads, buildings, bridges, ships, and etc. 5, some ones are naturel objects such as water bodies, coastlines, forestlands, planes, and etc [2][3]. Main purpose of object extraction process is to obtain useful information about interested object. A satellite object is generally composed of mosaic images. To capture an image of an entire object, image stitching preprocessing is needed.

With the launching different satellites during recent years, many different resources have been used to stitch satellite images. Sang [4] and Wahed et al. [5] and used QuickBird images. Manera et al. [6] and Cheng et al. [7] obtained their images from Unmanned Airborne Vehicles (UAVs) and used Tetracam images. Rube et al. [8], Wong and Clausi [9] and Fonseca and Manjunath [10] used LandSat-7 images for registering. Danchao et al. [11], and Fonseca and Manjunath [10] use SPOT-5 satellite images. Palmann et al. [12] used a radar-based system to obtain ENVISAT/ASAR images. Manera et al. [6] used IKONOS satellite images. Sayar et al. used LanSat-8 [13][14]. All aforementioned researches used certain number of formerly known mosaics to stitch them. In this paper, which mosaics to be used to obtain interested entire object is unknown previously. The proposed system is based on distributed architecture and selects mosaics automatically from mosaic dataset.

In the literature, a number of distributed and parallel computing based techniques have been proposed to process satellite images. Zhanfeng et al. [15] constructed a distributed processing system for processing remotely sensed images. Then, they performed several types of algorithms such as image segmentation, image classification, image target recognition, and etc. Ariel et al. [16] proposed a MapReduce model to solve two spatial problems: bulk-construction of R-Trees and aerial image quality computation on a Google & IBM cluster. The experimental results show that MapReduce can improve task completion times. Golpayegani and Halem [17] suggested parallel computing framework for satellite data processing of remote sensing systems. They implemented one month of gridding problem. Zhenhua et al. [18] introduced parallel k-means clustering of remote sensing images based on MapReduce programming model and they compared it with traditional one. Junfeng et al. [19] designed a remote sensing image service framework to providing static and dynamic web map service. Mamta et al. [20] reviewed recent development in high performance computing (HPC) technology for satellite data processing and analyzing. The related studies mentioned above use parallel, distributed and cluster computing technologies to process satellite images. However, no previous work is done for mosaic selection automatically with HPC technology to our knowledge.

The remainder of this paper is organized as following. Section 2 introduces the preliminaries on Apache Hadoop and Apache Lucene. Section 3 details with proposed models to select related mosaics for user defined region (spatial selection query) from mosaic dataset and evaluation scenarios. Section 4 concludes the paper.

## 2. PRELIMINARIES ON HADOOP AND LUCENE
This section explains the Apache Hadoop which is open source MapReduce implementation for large scale data processing in distributed manner on clusters of computers and HBase storage system. Moreover, Lucene technology providing scalable and high performance indexing, powerful, accurate and efficient search algorithms on cross platforms.

### 2.1 Hadoop
Hadoop is a distributed master-slave architecture that consists of the HDFS –scalable and reliable file system- for storage and MapReduce for computational capabilities. HDFS has two kinds of nodes: namenode (master) and datanodes (slaves). Namenode manages the file system namespace and stores metadata for all files and directories. Hadoop runs MapReduce jobs in parallel manner. To achieve processing of large amounts of data with MapReduce programming model, the developer has to define two functions: Map and Reduce. Input and outputs of these functions are records as <key, value> pairs. After users upload input data to the HDFS and

start jobs by implementing Map and Reduce functions, jobs are executed on worker nodes as MapTask or ReduceTask. Hadoop converts the input files into InputSplits and each task processes one InputSplit. InputSplit size should be configured carefully, because InputSplits can be stored more than one block if InputSplit size is chosen to be larger than HDFS block size (each block in the HDFS is 64MB by default). After all Map tasks are finished, their outputs are sorted and they become the input of the Reducer. In other words, the output format of the map function and the input format of the reduce function are same. Once the Reduce phase is finished and its output has been written back to HDFS, the user then retrieves the resulting data. The content of the records can be changed by implementing another derived class from RecordReader class [21][22]. Main advantages of Hadoop MapReduce framework are scalability, cost effectiveness, flexibility, speed, and resilience to failures [23].

HBase is a kind of NoSQL database software based on Hadoop [24]. A record of HBase table consists of a sole row key and some column families. Every column family has one or more columns defined by qualifiers. A programmer can access a table easily by APIs provided by HBase.

## 2.2 Lucene

Lucene is an open source full text search library. It has high performance and provides scalable information retrieval (IR). It also lets researchers searching capabilities to their applications built on top of Lucene. Lucene is an open-source project implemented in Java and concerns itself with text indexing and searching. The source of the data (files, web pages or documents), its format (XML, PDF, or HTML), or even its language are not important while indexing and searching [25].

Class explanations in Lucene's indexing are as following: Directory class stands for the directory storing index files; Document class stands for an indexing document, such as a text or a record of a table; Field class stands for different parts of the document and IndexWriter class is responsible for creating index for the document.

# 3. PROPOSED ARCHITECTURE

We propose two approaches to select mosaics from mosaic dataset. In both approaches, user firstly defines the corner coordinates of spatial object by dragging mouse on Google Map (see Figure 1). After selection process, related mosaics to be needed to capture the selected object are found by means of Hadoop based approach and Lucene based approach separately. All required mapping tools and services are provided by Google Map REST API. It is integrated to the system through Java framework and its standard open source libraries. A sample HTTP request to the Google's REST API for the map is given below:

```
http://maps.google.com/maps/api/staticmap?
```

After the question mark, query parameters with their user assigned values are appended. Parameter-value pairs are separated by ampersand (&) symbols. A sample query is given below:

```
http://maps.google.com/maps/api/staticmap?center=Kocaeli+University,+İzmit,+Turkey&zoom=14&size=512x512&maptype=roadmap
```


**Figure 1. An example of spatial selection query**

Details of proposed approaches and their effectiveness in terms of response time under varying number of mosaics are as given at following subsections.

## 3.1 Hadoop based mosaic selection approach

Figure 2 depicts the Hadoop based mosaic selection approach. All mosaics and their bottom left and upper right coordinates are stored in HBase. After user selects the region (query region) on Google Map, spatial region query could be resolved with one MapReduce job. This job includes one Map function only. In the Map function, the filtering strategy can be used to find the mosaics meeting the requirement of region query. The results of the Map stage are stored in the distributed file system directly [26]. In filtering phase, bottom left and upper right coordinates of every mosaic are examined whether intersects with the query region or not. Pseudo-code of filtering step is as following:

**Algorithm 1** Finding Intersected Mosaics

**Procedure** MosaicSelection (minX, minY, maxX, maxY)
**Input:** Parameters that indicates the starting (minX, minY) and ending (maxX, maxY) coordinates of a mosaic
**Output:** An intersected mosaic list

1. **begin**
2. **for each** (mosaic_i **in** mosaic dataset) **do**
   **if**((minX<=qminX)&&(maxX>=qminX)&&(minY<=qminY)&&(maxY>=qminY)) **then** mosaic_i is included into intersected mosaic list
   **if**((minX<=qminX)&&(maxX>=qminX)&&(minY<=qmaxY)&&(maxY>=qmaxY)) **then** mosaic_i is included into intersected mosaic list
   **if**((minX<=qmaxX)&&(maxX>=qmaxX)&&(minY<=qmaxY)&&(maxY>=qmaxY)) **then** mosaic_i is included into intersected mosaic list
   **if**((minX<=qmaxX)&&(maxX>=qmaxX)&&(minY<=qminY)&&(maxY>=qminY)) **then** mosaic_i is included into intersected mosaic list
3. **end for each**
4. **end**

where (qminX, qminY) and (qmaxX, qmaxY) show bottom left and upper right coordinates of query region, respectively. (minX, minY) and (maxX, maxY) show bottom left and upper right coordinates of mosaic_i in mosaic dataset, respectively.

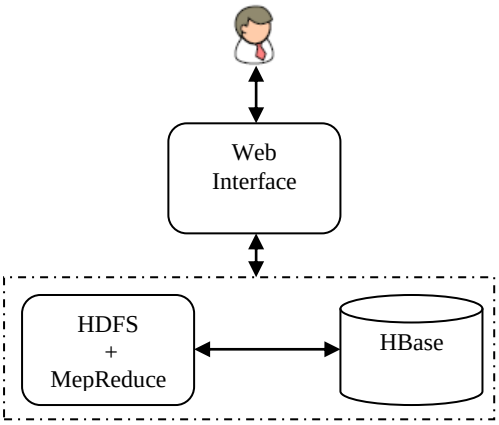


**Figure 2. Hadoop based mosaic selection architecture**

## 3.2 Lucene based mosaic selection approach

Figure 3 depicts the Lucene based mosaic selection approach. All mosaics and their bottom left and upper right coordinates are stored in input mosaic directory. All index files created by Lucene are stored in index directory. After user selects the region (query region) on Google Map, intersected mosaics with spatial region query are transferred to output mosaic directory. Intersection test is performed as Hadoop based approach.

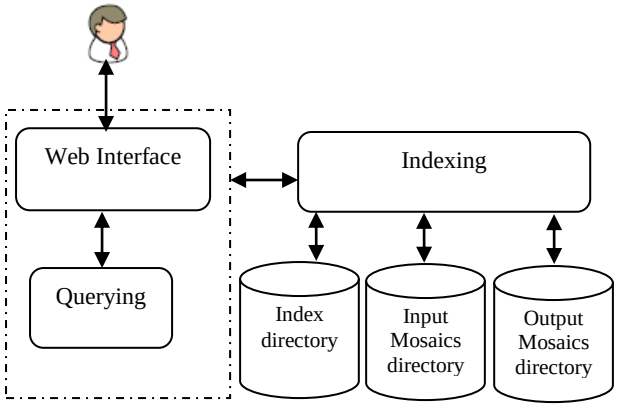


**Figure 3. Lucene based mosaic selection architecture**

## 3.3 Performance Evaluation

To evaluate effectiveness of the proposed approaches, their response times are obtained under varying number of mosaics. As if the time spent on region query is related to the size of region, because the large query window leads to large set of results from the filter stage. However, bottom left and upper right coordinates of every mosaic is used instead of size in proposed approaches. So, size of query region is not important in this study. Figure 4 illustrates response times for both implementations graphically. All experimental results are obtained on one machine.

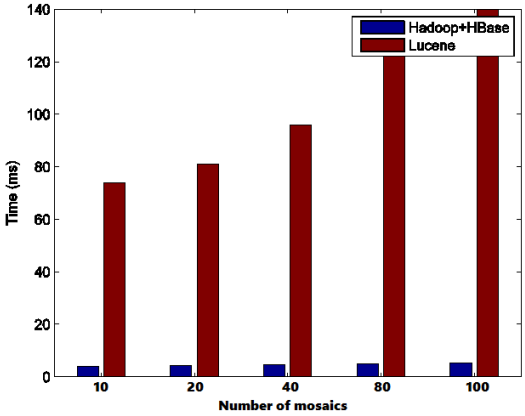


**Figure 4. Comparison of average processing times**

## 4. CONCLUSION AND FUTURE WORK

Object extraction from remote sensing images needs to register mosaic images. For user specified region, which mosaics to be included to registration process is critical problem. It can be seemed that these mosaics are known. In this study, we implement full automatic mosaic selection approaches. Former one is based on hybrid of Apache Hadoop and HBase and latter one is based on Apache Lucene. The performance evaluation demonstrates the feasibility of processing spatial selection queries with MapReduce. Compared with Lucene, spatial selection queries evaluation with MapReduce outperforms Lucene. The proposed approaches can be used in object extraction, object recognition, and image stitching as a preprocessing step.

In the near future, we plan to use more computing nodes to test both approaches in terms of horizontal scalability. Also, we will register interested rectangular sub regions (selected mosaics) using their vectorial information in distributed architecture. Because, we can say that if the number of pixel of the satellite image increases, it is necessary to have more memory space. Also, processing satellite images is harder than processing any other images. There are a number of reasons for this: bad weather and atmospheric conditions, color variations, satellite images from different sensors usually have different spatial resolution, and etc.

## 5. REFERENCES

[1] Zhongbin, L., Wenzhong, S., Qunming, W., Zelang M. 2012. Extracting Man-Made Objects From Remote Sensing Images via Fast Level Set Evolutions. IEEE Transactions on Geoscience and Remote Sensing, 53, 2, 883-899.

[2] Eken, S. and Sayar, A. 2015. An automated technique to determine spatio-temporal changes in satellite island images with vectorization and spatial queries. Sadhana, 40, Part 1, pp. 121–137.

[3] Eken, S. and Sayar, A. 2012. Vectorization and Spatial Query Architecture on Island Satellite Images. Procedia Comput. Sci. J. 2: 37–43.

[4] Lee, S. R. 2010. A coarse-to-fine approach for remote-sensing image registration based on a local method. International Journal on Smart Sensing and Intelligent Systems, 3, 690-702.

[5] Wahed, M., El-tawel, G. S., El-karim, A. G. 2013. Automatic Image Registration Technique of Remote Sensing Images. International Journal, 4, 177-187.

[6] Manera J. F., Rodrigez L., Delrieux, C., Coppo, R. 2010. Aerial image acquisition and processing for remote sensing. Journal of Computer Science & Technology, 10, 97-103.

[7] Cheng, Y., Xue, D., Li, Y. 2007. A fast mosaic approach for remote sensing images. In International Conference on Mechatronics and Automation, Heilongjiang, China, pp. 2009-2013.

[8] Rube, I. E., Sharkas, M., Salman, A., Salem, A. 2011. Automatic Selection of Control Points for Remote Sensing Image Registration Based on MultiScale SIFT. In International Conference on Signal, Image Processing and Applications, Chennai, India.

[9] Wong, A. and Clausi, D.A. 2007. ARRSI: automatic registration of remote sensing images. IEEE Transactions on Geoscience and Remote Sensing, 45, 1483-1493.

[10] Fonseca, L. M. G. and Manjunath, B. S. 1996. Registration techniques for multisensor remotely sensed imagery. PE & RS- Photogrammetric Engineering & Remote Sensing, 62, 1049-1056.

[11] Danchao, G., Xiaotao, T., Shizhong, L., Guojun, H. 2008. Image registration of high resolution remote sensing based on straight line feature. In International Society for Photogrammetric and Remote Sensing Symposium, pp. 1819-1823.

[12] Palmann, C., Mavromatis, S., Sequeira, J. 2008. SAR image registration using a new approach based on the generalized hough transform. ISPRS 2008-Beijing (Chine), XXXVII. Part B7, 145-152.

[13] Sayar, A., Eken, S., Mert, U. 2013. Registering LandSat-8 Mosaic Images: A Case Study on the Marmara Sea. In IEEE 10th International Conference On Electronics Computer and Computation, pp. 375-377.

[14] Sayar, A., Eken, S., Mert, U. 2014. Tiling of Satellite Images to Capture an Island Object. Communications in Computer and Information Science, 459, pp. 195–204.

[15] Zhanfeng, S., Jiancheng, L., Guangyu, H., Dongping, M., Weifeng, M., Hao S. 2007. Distributed computing model for processing remotely sensed images based on grid computing. Information Sciences, 177, 504–518.

[16] Ariel, C., Zhengguo, S., Vagelis, H., Naphtali, R. 2009. Experiences on Processing Spatial Data with MapReduce Scientific and Statistical Database Management, Lecture Notes in Computer Science, 5566, 302-319.

[17] Golpayegani, N., Halem, M. 2009. Cloud Computing for Satellite Data Processing on High End Compute Clusters. In IEEE International Conference on Cloud Computing, pp. 88-92.

[18] Zhenhua, L., Yingjie, H., Haidong, Z., Jianping, W., Bo, L., Hui, Z. 2010. Parallel K-Means Clustering of Remote Sensing Images Based on MapReduce. LNCS 6318, pp. 162–170.

[19] Junfeng, K., Zhenhong, D., Xiaosheng, L. 2012. The Framework of Remote Sensing Image Map Service on Hadoop. In National Conference on Information Technology and Computer Science, pp. 868-869.

[20] Mamta, B., Abhishek, C., Anshu, P., Sivakumar, V. 2013. High Performance Computing for Satellite Image Processing and Analyzing – A Review. International Journal of Computer Applications Technology and Research, 2, 4, 424-430.

[21] J. Dean and S.Ghemawat, "MapReduce: simplified data processing on large clusters," Communications of the ACM, (2008) 51(1): 107–113.

[22] Official Hadoop Web Site, 2015, http://hadoop.apache.org/. (2015).

[23] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., Stoica, I. 2010. Spark: cluster computing with working set. In Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing, pp. 1–7.

[24] Lars G. 2011. HBase: The Definitive Guide, Sebastopol, CA: O'Reilly.

[25] Erik H., Otis G., Michael M. 2009. Lucene in Action, Manning Publications.

[26] Shubin, Z., Jizhong, Han., Zhiyong, L., Kai, W., Shengzhong, F. 2009. Spatial Queries Evaluation with MapReduce. In Eighth International Conference on Grid and Cooperative Computing, pp. 287-292.