Acta Cybernetica **22** (2015) 57–79.

# A Case Study of Advancing Remote Sensing Image Analysis[*]

Roberto Giachetta[†] and István Fekete[†]

**Abstract**

Big data and cloud computing are two phenomena, which have gained significant reputation over the last few years. In computer science the approach shifted towards distributed architectures and high performance computing. In case of geographical information systems (GIS) and remote sensing image analysis, the new paradigms have already been successfully applied to several problems, and systems have been developed to support processing of geographical and remote sensing data in the cloud. However, due to different circumstances many previous workflows have to be reconsidered and redesigned.

Our goal is to show a way how the existing approaches to remote sensing image analysis can be advanced to take advantages of these new paradigms. The task aiming in shifting the algorithms shall require a moderate effort and must avoid the complete redesign and reimplementation of the existing approaches. We present the whole journey as a case study using an existing industrial workflow for demonstration. Nevertheless, we define the rules of thumb, which can come in hand when shifting any existing GIS workflows.

Our case study is the workflow of waterlogging and flood detection, which is an operative task at the Institute of Geodesy, Cartography and Remote Sensing (FÖMI). This task in currently operational using a semi-automatic single machine approach involving multiple software. The workflow is neither efficient nor scalable, thus it is not applicable in emergency situations where quick response is required. We present an approach utilizing distributed computing, which enables the automated execution of this task on large input data with much better response time. The approach is based on the well-known MapReduce paradigm, its open-source implementation, the Apache Hadoop framework and the AEGIS geospatial toolkit. This enables the replacement of multiple software to a single, generic framework. Results show that significant performance benefits can be achieved at the expense of minor accuracy loss.

**Keywords:** remote sensing image analysis, object based image classification,

[*]This research is founded by the FP7 project IQmulus (FP7-ICT-2011-318787) a high volume fusion and analysis platform for geospatial point clouds, coverages and volumetric data set.

[†]Eötvös Loránd University, Faculty of Informatics and Institute of Geodesy, Cartography and Remote Sensing E-mail: {groberto,fekete}@inf.elte.hu

DOI: 10.14232/actacyb.22.1.2015.5

flood and waterlogging detection, cloud computing, MapReduce, Hadoop

## 1 Introduction

In the last decades serious researches were carried out advancing the algorithmic capabilities resulting in the evolution of parallel and distributed processing. A great attention has been paid to these improvements leading to the increasing popularity of distributed and high performance computing (HPC). In parallel, cloud computing [14] has become a mainstream method. In the field of IT, professionals accepted the challenge, therefore, nowadays information systems present virtually unlimited possibilities for data analysis.

Due to data sets becoming increasingly large and complex (usually noted as *big data*), these new paradigms cannot be ignored [1]. However, to take advantage of todays state of the art computing, previous data processing methodologies and workflows have to be revisited and redesigned. In addition, the overwhelming amount of data leaves less room for user interaction and requires more automation on behalf of the processes.

The new paradigms have also reached geographical information systems (GIS) and remote sensing image analysis leading to the development of *spatial cloud computing* [50]. Multiple solutions have been presented in many areas utilizing cloud technology for processing big spatial and remote sensing data. These solutions present a dedicated approach to a specific environment [2, 19, 32]. In contrary, most spatial data analysis processes performed at organizations such as the *Institute of Geodesy, Cartography and Remote Sensing* (FÖMI) have their evolved workflows using multiple (proprietary and open-source) software and GIS expertise [13, 34]. Most tasks are semi-automatic, involving some manual adjustments and fine tuning and rather work with files instead of databases.

To demonstrate the effort required for such advancement, the workflow of *flood and waterlogging detection* was chosen as case study. We highlight the main obstructions here. First, this process consists of several steps including interaction from a remote sensing expert. Note that this fact avoids the automation. Second, as high-resolution satellite imagery is used as input data, in some cases the size of input is too large to be handled by a single machine. Last, an important aspect is that sometimes the results are required within a limited time frame. To satisfy all criteria, a distributed, scalable approach embedding full automation and also providing high performance is required. Hence, the advancement of the workflow requires both architectural and algorithmic considerations. In this paper, a solution that meets the above described criteria is presented.

The main contributions of the paper are

- the study about how our previously researched methodology was put into practice,
- the introduction of a method for adapting object based thematic classification in a distributed environment, and

- the adequate evaluation of the advanced workflow highlighting accuracy and performance capabilities.

The rest of the paper is arranged as follows. Section 2 presents the case study by describing the current solution for flood and waterlogging detection, requirements and project objectives. Section 3 presents related work and background of the project. Section 4 details the proposed solution from both algorithmic and architectural point of view. Section 5 presents performance and accuracy results. Section 6 concludes the paper.

## 2 The process of flood and waterlogging detection

Waterlogging and flood detection is an operative task at FÖMI [12]. As part of the disaster recovery project it helps to estimate the measure of logging damage and foresee the time of water withdrawal.

Sometimes the operation is performed in emergency cases, where quick reaction is required [8]. Although real-time evaluation is not necessary, to enable decision makers to react based on up-to-date information the process must be performed in a limited time frame. Thus, the operation cannot take more than a few minutes, and should not require user interaction (as an expert may not be available at the specified time, or cannot respond with the required speed). As both floods and waterlogging may occur over large areas, multiple high resolution aerial and satellite imagery must be processed within the time frame, requiring either high performance or parallel computation.

As it can be seen, the process must satisfy multiple criteria with respect to performance and execution time to be useful in emergency situations. Unfortunately, the current solution, which relies on supervised classification techniques cannot satisfy the above mentioned conditions.

### 2.1 Supervised classification

The currently used flood detection method is based on satellite image analysis primarily using *SPOT 5* and *Landsat 8* imagery. The solution program is running mainly in the *ERDAS Imagine* environment but additional custom tools are also required.

The workflow consists of two steps.

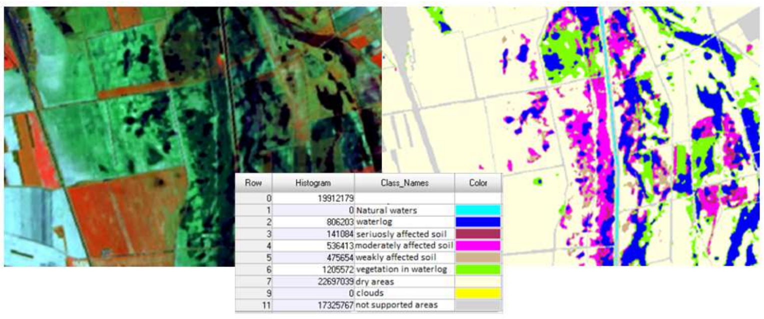
1. *Preprocessing of satellite images*, including: image registration and geometric corrections (only required on raw images), cloud and cloud shadow filtering for producing a cloud mask, computation of the Top of the Atmosphere (ToA) reflectance and spectral indices.

   Cloud and cloud shadow masking is an important step in the preprocessing phase, because cloud shadow and water can be spectrally similar. ToA reflectance [18] represents the solar radiation incident on the instrument in standard unitless terms, independent of the position of the sun with respect to the earth. Spectral indices [33] are designed to convert spectral reflectance into biophysical information that can be interpreted directly by the user. Three indices are used for the process, namely *vegetation* (NDVI), *soil* (NDSI) and *water* (NDWI) that can be computed based on the red, near infrared and short-wave infrared bands of the source image. ToA reflectance and spectral indices can be computed automatically for each pixel if proper satellite metadata is available.

2. *Classification of the images* based on multiple input data, namely the calibrated (ToA reflectance) image, the spectral indices, the cloud mask and the mask of natural waters (rivers and lakes).

   The classification process uses multiple thresholding operations based on computed parameters (15 threshold values), and other parameters specified by the remote sensing expert. These values are not predefined and not precomputed, but adjusted by examining the result of the classification. Also, the parameters have to be re-set for each image, and unique starting values are used for different satellites. Thus, the processing time of a set of images relies mainly on the manual recalibration time, and can take multiple hours. The thresholding is also pixel based, without taking any information of neighboring cells into account.

The process results a thematic map with the following categories: natural waters, waterlogging, seriously affected soil, moderately affected soil, weakly affected soil, vegetation in water, dry areas, clouds and not supported areas. Figure 1 shows an example.



| Row | Histogram | Class_Names | Color |
|---|---|---|---|
| 0 | 19912179 | | |
| 1 | 0 | Natural waters | 🟦 |
| 2 | 806203 | waterlog | 🟦 |
| 3 | 141084 | seriuosly affected soil | 🟥 |
| 4 | 536413 | moderately affected soil | 🟪 |
| 5 | 475654 | weakly affected soil | 🟫 |
| 6 | 1205572 | vegetation in waterlog | 🟩 |
| 7 | 22697039 | dry areas | ⬜ |
| 9 | 0 | clouds | 🟨 |
| 11 | 17325767 | not supported areas | ⬜ |

Figure 1: Flood and waterlogging detection

The final result of the analysis can be ranked as excellent but this quality is obtained in an elaborated and cumbersome time consuming way. Although utilizing human expert knowledge has benefits with respect to accuracy, the current solution relies on manual intervention beyond the optimal measure. The complete algorithm is presented in Algorithm 1.

**Algorithm 1** Supervised classification algorithm.

```
Funct    Classification($image, metadata, image_{water}, th_1, \ldots, th_n$)

1: $image \leftarrow$ Preprocess($image$)    Perform required preprocessing
2: $image_{cloud} \leftarrow$ MaskClouds($image$)    Compute cloud mask
3: $image_{class} \leftarrow$ Image($image$)    Create the classified image
4: for all $pixel \in image$ do
5:     $pixel_{toa} \leftarrow$ ToARef($pixel$, $metadata$)    ToA reflectance using metadata
6:     $pixel_{index} \leftarrow$ SpIndex($pixel_{toa}$)    Spectral indices based on reflectance
7:     if $pixel \in image_{water}$ then
8:         $pixel_{class} \leftarrow 1$    Delineate natural waters
9:     else
10:        if $pixel \in image_{cloud}$ then
11:            $pixel_{class} \leftarrow 9$    Delineate clouds
12:        else
13:            $pixel_{class} \leftarrow$ Th($pixel_{toa}, pixel_{index}, th_1, \ldots, th_n$)
14:                Threshold all categories
15:        end if
16:    end if
17: end for
18: return $image_{class}$
```

## 2.2 Considerations for enhancement

As seen in Section 2.1 there is much room for improving the current solution to be usable in all cases, especially in emergency scenarios. At the beginning of the project, the following considerations were made.

- The solution must handle the analysis targeting large areas within a valid time frame. The input data may consist of several high-resolution satellite images, the size of which fall between a hundred megabytes and a few gigabytes. The entire dataset may have a size up to multiple terabytes. Additional data may also be provided in the future, such as aerial imagery or digital elevation models (DEM).

- The process must enable fully automated execution of the workflow. Limited compromise can be made with respect to accuracy for performance benefits. However, manual corrections and fine-tuning must also be permitted in the event of more accuracy and less performance is required.

- As separate areas (images) can be analyzed individually, the process can be parallelized to speed up the workflow.

- The entire process should be executed by a single software and not multiple applications. As the current solution relies on different software, data must be loaded and saved multiple times causing a performance bottleneck. Also, proprietary software should be eliminated from the process.

With respect to the execution environment, a widespread cloud computing platform is required due to multiple reasons.

- The platform should scale with the amount of data.
- Available commodity hardware can be integrated into the platform and can be utilized depending on additional usage.
- External providers (e.g. Amazon S3) can be resorted to if the available hardware is not capable of handling the input data within a valid time frame.

For these reasons, the industry standard Apache Hadoop [6] was chosen as execution environment. As detailed in Section 3.1 many experiments have been performed on the Hadoop platform and the project has gained many contributions, making it the most mature solution. Although there are many popular extensions to the basic Hadoop environment, such as Spark [52] and Hive [47], these additions are not specifically designed for image processing, and allow less room for customization, thus they are not considered. Hence, the data is stored in the *Hadoop Distributed File System* (HDFS) [45] and the algorithms are executed using the MapReduce model, which can be adapted easier to handle heterogeneous image data and function with external tools [49]. Certainly, the choice of environment has its advantages and drawbacks [37, 46], which one must cope with.

Obviously, HPC and GPU acceleration would be alternative approaches, which have their own history in remote sensing image analysis [36]. However, simply relying on multiple physical GPGPU machines would not enable scalability. Certainly, GPU acceleration can also be utilized in combination with cloud computing by allowing the local processes to run on GPU instead of the CPU [43]. This technique has been successfully applied in distributed image processing with respectful performance gains [42]. It requires the proper graphics accelerators to be available, but it does not influence the architecture of the approach, only the implementation of the local executables.

## 3 Background

Many research has been performed on moving geographical information systems into distributed environment to enable processing of big geospatial data [29, 30]. There are some experiments in distributed remote sensing image analysis, which is briefly discussed in this section. Moreover, a detailed description of the AEGIS geospatial framework is presented.

## 3.1 Related studies on distributed image processing

Experimentation on distributed image processing has been performed for many years [23] both in terms of algorithmic solutions [39] and data storage and retrieval [26].

When cloud computing and the MapReduce model became prominent, many fields of computer science started shifting towards the new paradigm [25]. Unfortunately, as MapReduce was primarily developed for processing simple text documents [15], handling of complex binary structures such as image formats was not considered in early years. However, due to advancements Hadoop has become a frequently used image processing platform [4].

Golpayegani and Halem showed that implementing image analysis operations in Hadoop can be performed in a straightforward way [24]. By using a cell based allocation of images, the *Map* phase is used both as a locator and initial processor, whilst the *Reduce* phase performs result summation. A different approach to distributed image processing is presented by Alonso-Calvo et al. [3]. In this study, images are transformed to region-based graph representation allowing operations to work on the distributed regions. Although this method enables easy parallel execution of most image processing operations, the region based transformation may cause loss of information. Multiple studies also deal with image storage and retrieval techniques [44, 51, 53]. Stand-alone tools have also been developed in multiple fields, for example ballistics [31] and medical imaging [5].

In terms of remote sensing imagery the Hadoop platform has proven to be useful in case of image classification. Li et al. presented a variant of parallel ISODATA clustering that is specifically designed for the MapReduce scheme [38]. Maximum Likelihood classification has also been ported to MapReduce [48] in a two phase process, in which the Map phase performs sampling, and the Reduce phase is responsible for classification. Another approach to classification is presented by Codella et al. by using image matching and machine learning to extract land-cover information using multiple MapReduce operations [11]. Although working on satellite imagery, this methodology only utilizes the visible bands of the spectral data.

It can be seen that the Hadoop framework is a potential execution environment for many cases of remote sensing image analysis using commodity hardware. Studies showed that a complete image management and processing framework can be built on the top of Hadoop platform using standardized technology and open-source software [9, 27, 40].

However, as a flexible solution is required, the algorithms cannot be directly developed for Hadoop and MapReduce, but they should follow a more broad approach to enable their reuse on other platforms. Therefore, a generic framework is required that offers the possibility to execute operations in Hadoop, but is not specifically designed for the distributed MapReduce model. Such a general approach is presented by the AEGIS framework.

## 3.2 The AEGIS geospatial framework

The AEGIS framework [21] is a geospatial toolkit developed within a running R&D project at *Eötvös Loránd University*, *Faculty of Informatics* (ELTE IK), which aims to research new approaches in geospatial data management. It supports several kinds of geospatial data, including vector datasets, raster imagery and point clouds. It is a platform independent library, implemented using *.NET/Mono Framework* to exploit the wide possibilities and the simple usage of this object-oriented development platform.

AEGIS has been developed by taking adaptability and extensibility in mind. It employs state of the art programming methodologies and contains possible realizations of well-known standards of the *Open Geospatial Consortium* (OGC). The component-based infrastructure enables the separation of working fields and the interchangeability of data models, methods and algorithms. The extensibility also enables marrying AEGIS with existing toolkits after the implementation of the proper wrappers for processing environment and data model.

All geospatial data, including raster imagery is considered a form of *geometry*, as defined by the *Simple Feature Access* (SFA) standard [28]. Algorithms are also handled in a uniform manner using metadata for describing operation methods. The metadata enables AEGIS to validate and optimize execution of methods and catalog them. New methods may be added, or existing methods can be extended to support new functionality or input data. Methods can have different representation and different restrictions in terms of operations. For example, Lanczos resampling [41] of an image may be performed using interpolation formula or using a precomputed table. These two variations can become operations of the Lanczos method with the latter used in imagery with small radiometric resolution. If not specified directly, AEGIS is responsible for selecting the appropriate operation for the specified method and input data.

AEGIS also supports data management and processing within Hadoop [22]. The framework enables the handling of complex binary input data in Hadoop, such as the GeoTIFF format, including image content and attributes. Input data may also be partitioned based on multiple properties to enable parallel processing of a single source on multiple nodes. Operations can be execution as Map or Reduce functions within the MapReduce process without any alteration. This approach does not require reimplementation of the operations, it simply relies on the replacement of the execution environment. However, this approach only applies to operations, where local computations yield the final result. In case of regional or global methods, the operation must be properly investigated whether its application on the individual parts yields the same result. If not, the proper alternative must be specified and implemented. The alternative may come in the form of the original operation and some additional post-processing method, thus not requiring major effort.

For example, histogram equalization is a global operation, which requires the complete image histogram. Applying equalization over image parts can result to completely different outcome. Hence, equalization is applied in three steps in the

MapReduce environment. The first step computes the histogram of the individual image parts in the Map phase. The second step merges the histograms and computes the mapping of the values in the Reduce phase. The third step applies the mapping to the image parts in a second Map phase, creating the final result. The overview of the operation can be seen in Figure 2.

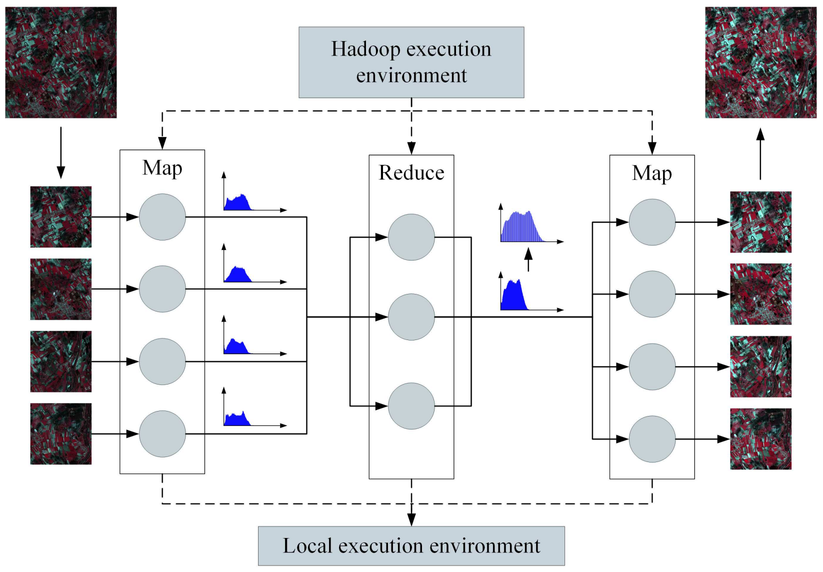


Figure 2: Histogram equalization performed in MapReduce

In summary it can be said that although moving operations to Hadoop based execution can be performed easily in many cases using the AEGIS framework, in some cases it cannot be done automatically. Additionally, the adaptation of the algorithm may have impact on the allocation of data on the distributed file system, as it can be seen on the case study, described in Section 4.2.

## 4 Advancing flood and waterlogging detection

To meet the requirements defined in Section 2.2, both automation and cloud based distributed execution are required. Automation can be acquired by replacing the supervised classification procedure (e.g. image thresholding) with unsupervised classification. This approach is detailed in Section 4.1, whilst the application of the workflow in cloud environment is described in Section 4.2.

### 4.1 Shifting to unsupervised classification

Unsupervised classification is performed by providing a *reference dataset* of manually classified areas, which are divided into two disjoint parts, namely *training areas* and *test areas*. Thus, the input image is first clustered (using *ISODATA* clustering with *Bhattacharyya* distance [10]), then spectral properties of the clusters are matched to the training area.

Sets of reference data are available from previous results obtained by using the semi-automatic approach presented in Section 2.1. As these results are fine-tuned manually, they provide a proper set of spectral data to be used by the classifier. However, not the entire dataset is used as reference data as spectral properties and environmental conditions of the reference and the current images may vary. Reference data for the specific input is selected based on image attributes, such as sensor type, imaging time and geographic location.

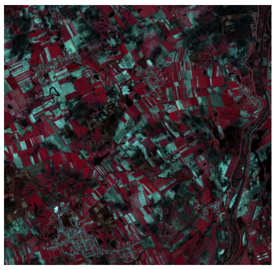
With respect to accuracy, previous research [35] has shown that *object-based thematic classification* [7] yields better results in most cases than traditional pixel based methods. Therefore, the process is enhanced by prior segmentation of the input data. The application of segmentation is not only an option, but a necessity in the processing of high resolution images, as their pixels usually cannot be interpreted individually. The segmentation results in image objects, representing a contiguous set of spectrally similar pixels. There is a wide range of segmentation methods available. The examined methods include *sequential linking*, *best merge*, *graph-based merge* and *quadtree based segmentation*. These algorithms were examined in previous experiments with object based image classification [16]. The algorithms have multiple parameters, that influence the resulting quality and number of segments.

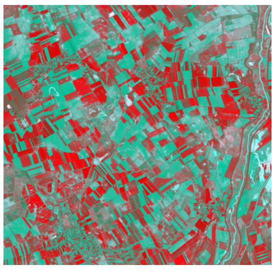
The process also includes automatic evaluation of the result based on the test area, for which a *confusion matrix* is computed. The number of correctly categorized pixels with respect to the size of the test area results in the accuracy percentage.

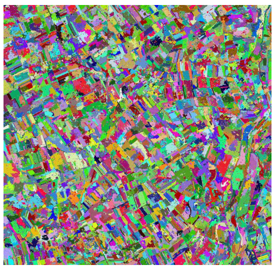
Certainty, accuracy is significantly affected by parameters of the segmentation and clustering methods, which may vary for each input image. The adjustment of parameters can be performed by evaluating the confusion matrix, and re-executing the algorithms with a different set of parameters. *Simulated annealing* is used for fine tuning parameters starting from a set of baseline values. The fine-tuned parameter values are then stored in a catalog indexed by image properties and statistics (e.g. location, time, mean and variance of values). When new input data is processed later on, the process selects fine-tuned parameters from the catalog if the image properties can be matched. This allows the workflow to use more optimized parameters without re-execution. Obviously, these values can be optimized even further.

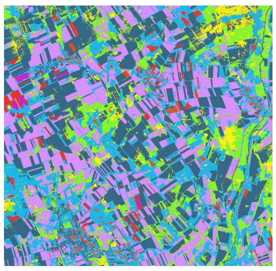
Unsupervised classification thus replaces the thresholding operations described in Section 2.1, as the final step(s) of the process. The approach also eliminates cloud filtering from the process, as it can also be performed within the classification. The revised workflow can be seen in Figure 3.

Although this attitude removes the possibility of manual corrections from the process, manual correction of the reference data is still possible, and even the final step of the operation can be reverted back to threshold based classification if needed.


(a) Input image


(b) Spectral indices


(c) Segmentation


(d) Clustering

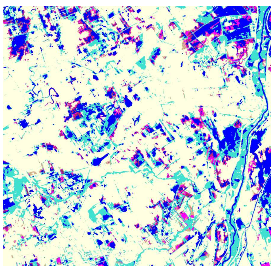

(e) Classification

Figure 3: Object based thematic classification applied to the case study

## 4.2 Workflow execution in Hadoop

In the Hadoop environment, processing is executed as a sequence of MapReduce operations, consisting of *Map* and *Reduce* phases. Both may have multiple instances executing on different nodes, identified by a *key*, which is propagated by the data. Data is shuffled between the phases by Hadoop and cached using a built-in caching mechanism. Data is allocated in HDFS in blocks, with adjustable maximum size, and each block is processed in parallel.

The simplest approach to Hadoop is the individual processing of input images by assigning unique keys to them. Each input image is loaded to HDFS in one block, and is processed independently. Thus, the execution of the entire workflow can be performed as a single Map phase. Unfortunately, due to the many transformation steps applied within the process (e.g. segments and clusters are created), the analysis of large files may cause performance issues due to memory restrictions.

A more general approach is to partition images into multiple, individually processable blocks, which can be performed by AEGIS beforehand. However, in order to determine the appropriate partitioning methodology, one must examine the properties (input, working set and output) of the algorithms. Consider the following.

- ToA reflectance and spectral index computation are local operations, and as such the partitioning of images does not influence their computation, as long as each pixel component is available locally. As such, both can be computed in a single Map phase as part of the preprocessing.

- Segmentation is a regional image operation, where spatially neighboring pixels are required for evaluation. The examined regions are of irregular shape, and may be different for each algorithm and even each configuration of an algorithm. The result of the segmentation is a segment map, which can be represented by a set of pixels for each segment.
- Clustering is also a regional operation, but in the multispectral space domain of the image, where the location of pixels (spectral vectors) is independent of their spatial location. Either the original image or the segment map can be specified for input, and the output is a cluster map (similar to the segment map).
- The reference based classification relies on the matching of the reference image to the result of the classification. It is also performed on pixel level.

One must observe that for segmentation the image cannot be partitioned to distinct parts, as neighboring pixels may influence the outcome. To counteract this, a *buffer zone* should be created, and data located near tile borders must be included in both tiles, as illustrated in Figure 4. From the point of segmentation, the size of the buffer area should be chosen so that generally segments fit into the buffer zone. This results in duplicate segments for each region within the buffer zone, which must be eliminated before clustering.

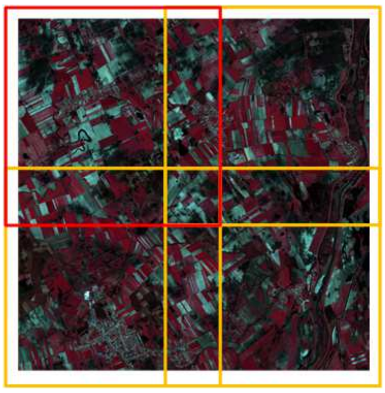


Figure 4: Tiling an image into 4 parts with buffer zones

Elimination of the duplicate segments can be performed using the following methodology.

- For matching segments (with the same set of pixels) one is removed.
- For segments that are subsets of another segment, there are two options. If the segment is on the border of the buffer zone, it is removed. If the segment is inside the buffer zone, it is kept and its pixels are removed from the greater segment.

- For segments which partially overlap, the overlapped region is removed from both to form a new segment.

The presented methodology results in generally more segments, but does not allow the causeless merging of any segment. As segmentation will be followed by clustering, the segments will be merged together anyway if spectral properties are similar. This post-processing of segmentation requires the pixels of the buffer zone and the segments which are located within the zone, and can be performed in parallel for each area within the buffer zone. For example in Figure 4 five buffer zones can be processed in parallel. The process results in the segmentation performed using both a Map and a Reduce phase.

It can be seen that spatial partitioning with buffering is a good choice for segmentation. The resulting segments can be forwarded to clustering. As the ISODATA method uses statistical properties of the segments (e.g. mean, variance, covariance) instead of pixel data, the amount of data required for clustering is generally less, and moving this data between nodes does not yield significant performance issues. Thus, the statistics of the resulting segments are shuffled based on their location within the multispectral space as key.

Clustering is performed in two steps. First, distinct parts the multispectral space are processed (using a distance metric, such as Bhattacharya distance). The number of parts can be the same as the number of tiles within the image. Second, clusters located near the border of a part are post-processed, and merged when required (using the same metric). This yields the final clusters, which are forwarded to each tile of the original image to create the cluster map of the tile. Again, the algorithm can be performed using a Map and a Reduce phase.

As for classification, the training area is matched against the cluster map. As the spatial location of the training data varies, the computation of class mapping is performed on the appropriate tile(s). The mapping is then forwarded to all tiles, creating the classification map. Classification requires a Map phase for matching, a Reduce phase for combining the matches, and another Map phase for applying the matches to all tiles. Hence, the algorithm is finished after five MapReduce processes (some of which lack the Reduce phase). The complete algorithm is presented in Algorithm 2.

For accuracy testing, the test area is applied in the same manner, and the confusion matrices of all tiles are aggregated for accuracy measure. Hence, accuracy measure requires an additional MapReduce process.

The presented approach is generalizable with respect to multiple input files (e.g. satellite image strips of an area). Hence, not only the processing of the individual images can be managed in this manner, but also the combined processing of all images. Segments can be adjusted on overlapping areas, whilst clusters can be adjusted uniformly for all images. This approach may also contribute to accuracy.

To summarize, three different workflow concepts are available.

- Individual processing of images and image tiles. Each process can be run in parallel in a single Map phase, and there is no communication between the processes.

**Algorithm 2** Distributed unsupervised classification algorithm.

```
Funct Classification($image, metadata, reference_{train}$)
1: for all $tile \in image$ do
2:   $tile \leftarrow$ map(Preprocess, $tile$)          Perform required preprocessing
3:   $tile \leftarrow$ map(Segmentation, $tile$)        Perform segmentation of tiles
4: end for
5: for all $zone \in$ Bufferzone($image$) do
6:   $segments \leftarrow$ reduce(Eliminate, $zone$, SegmentsOfZone($segments$))
7:                                     Perform elimination of duplicate segments
8: end for
9: for all $tile \in image$ do
10:   $clusters \leftarrow$ map(Clustering, SegmentsOfTile($segments$))
11:                                                  Perform clustering of tiles
12: end for
13: for all $nearClusters \in$ NearClusters($clusters$) do
14:   $clusters \leftarrow$ reduce(MergeClusters, $nearClusters$)
15:                                              Merge clusters near tile borders
16: end for
17: for all $tile \in image$ do
18:   $classMatch \leftarrow$ map(Classify, $tile, reference_{train}, clusters$)
19:                                  Perform classification based on training data
20: end for
21: $classification \leftarrow$ reduce(Combine, $classMatch$)
22:                                                   Combination of the matches
23: for all $tile \in image$ do
24:   $tile_{class} \leftarrow$ map(Apply, $classification$)
25:                                          Apply the classification to the image
26: end for
27: return $image_{class}$
28:                     The classified image is the combination of classified tiles
```

- Individual processing of images, but joint processing of image tiles of a single image. This results in communication between the corresponding processes in case one or more images are tiled.
- Joint processing of all images. This results to wide range communication between the individual MapReduce processes independently of image tiling.

## 5 Evaluation

The advanced flood and waterlogging classification method is evaluated based on both algorithmic and architectural aspects. First, accuracy is measured with respect to results of the supervised classification method presented in Section 2.1.

Then, performance of the implementation is measured in cloud environment.

## 5.1 Algorithmic accuracy

As described in Section 4.1, the revised workflow relies on unsupervised object based thematic classification. First, segmentation is performed on the image using one of the specified algorithms, the are clustered (using the ISODATA method), and finally, classification is performed based on the chosen training areas.

Accuracy of the results is measured by evaluating the categories on the test area using the confusion matrix. Multiple factors are involved that influence the accuracy of the algorithm, such as the selection of segmentation method, the optimization of parameters (using simulated annealing) and the selection of reference areas. The reference area is selected from the classification result of the supervised classification. As the supervised result is fine tuned for accuracy, it is a proper choice for evaluation.

As described in Section 4.1, parameters are set to a baseline value, that was determined by experimentation with multiple datasets. The baseline parameters can be seen in Table 1. Later, parameters are fine tuned for each image, resulting in near optimal values.

| Method | Parameter | Value |
|---|---|---|
| Sequential linking segmentation | Homogeneity criteria ($C_H$) | 0.03 |
| | First ANOVA threshold ($C_1$) | 0.005 |
| | Second ANOVA threshold ($C_2$) | 0.01 |
| Best merge segmentation | Number of iterations | 15 |
| | Merging threshold ($T$) | 10 |
| Graph-based merge segmentation | Scale of observation ($k$) | 250 |
| Quadtree based segmentation | Minimum object size ($mos$) | 0.015 |
| | Minimum quad size ($mqs$) | 2 |
| | Homogeneity threshold ($ht$) | 0.2 |
| ISODATA clustering | Initial number of cluster | 1000 |
| | Maximum distance from center | 0.8 |

Table 1: Baseline parameters for the individual methods.

A total of 30 Spot 5 images were tested in different scenarios with over 80 test runs. All images were classified beforehand using the supervised workflow. The results are visible in Table 2. Accuracy is presented with respect to baseline values for all images and the best result gained by optimizing parameter values for a single image. In the baseline case minimum, maximum and mean values are listed.

Previous experiments showed that there is no segmentation algorithm that performs best in all scenarios [17]. This was reflected in the current test as well. The accuracy of the segmentation method is highly influenced by the optimization of the parameters. Based on the mean and variance of the results, graph-based segmentation [20] performed best in most cases. Also, this method seemed to be less

| Segmentation method | Baseline | | | Optimized |
|---|---|---|---|---|
| | Mean | Min | Max | Max |
| Sequential linking | 79.6 | 53.2 | 88.9 | 94.3 |
| Best merge | 81.9 | 71.4 | 87.1 | 88.5 |
| Graph-based merge | 87.7 | 81.3 | 91.2 | 93.8 |
| Quadtree based | 84.5 | 66.9 | 91.9 | 92.3 |

Table 2: Accuracy results (in percentages) with respect to parameter optimization.

effected by the individual image conditions.

It must also be noted that some differences were found depending on the tiling of the image. Executing the workflow on the tiled image without any buffer area resulted up to 13.2% less accuracy than processing with 200 pixels of buffer area.

Finally, the workflow using graph-based merge segmentation was selected for evaluation in joint processing of the entire dataset. Using this approach the average accuracy for baseline parameter values was raised to 90.2%, over 2% more than by separate processing of images. Although this value is still less than the accuracy in case of optimized parameters, the workflow executed much faster, as no multiple iterations are required for fine tuning of parameters.

## 5.2 Performance in the cloud

The performance of the revised workflow was measured in a Hadoop cloud consisting of 16 virtual machines. The configuration of the machines included a single thread 3 GHz processor and 8 GB system memory. Although the number of machines is far less than the size of a common computational cloud, this set of nodes enables the examination of system behavior in processing a single high resolution remote sensing image partitioned into multiple tiles. As each image is processed individually (when not performing joint processing of all images), the combined performance of the entire cloud will not differ greatly from the case of processing a single image. Naturally, when working with a large collection of source images a greater cloud is required.

The processing time of a single image is dependent on multiple factors, including the segmentation method and parameter fine tuning. Images of three different size have been tested to evaluate the behavior of individual methods with respect to image size. The dataset includes small ($1920 \times 1780$), medium ($3940 \times 3690$) and large ($7880 \times 7380$) size SPOT 5 satellite imagery. Also, images are partitioned up to 16 tiles. Thus, the workflow of an image partitioned into $n$ tiles is different than the workflow for processing $n$ individual images, as it is performed in multiple MapReduce functions.

Figures 5, 6 and 7 contain the results of performance evaluation of execution with different image size and number of tiles. Note, that these results only contain the execution time of classification (e.g. MapReduce jobs), and do not contain the time of Hadoop job initialization and the partitioning of images. As partitioning is performed during the allocation of the image in HDFS, the additional execution time is only marginal in comparison with the uploading of the image using the standard HDFS access channels.

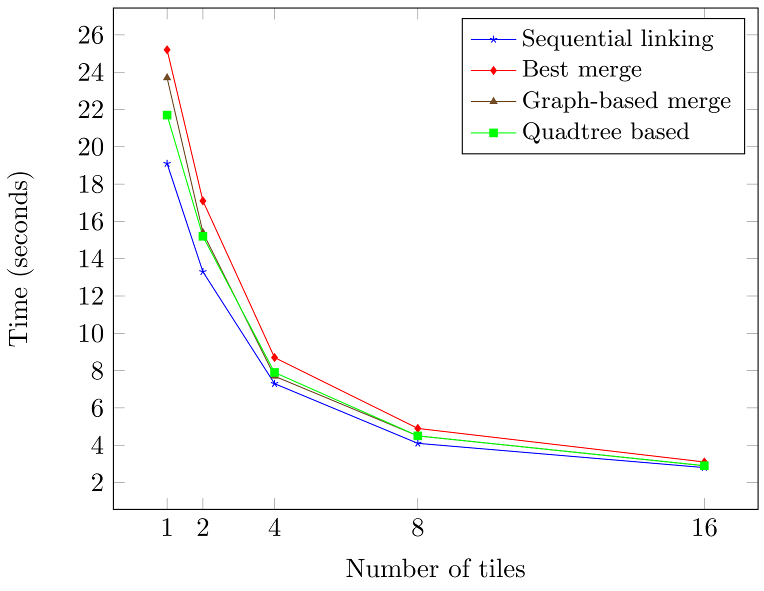


Figure 5: Average execution time on small images

As expected, the selection of segmentation method greatly influences execution time of the algorithm. Generally sequential linking performs the fastest, whilst graph-based merge segmentation is the mostly influenced by image size, and it takes the most time win case of large images.

Naturally, partitioning the image into multiple tiles greatly improves performance. The additional processing time required for post-processing (e.g. elimination of duplicate segments, data shuffling) is more noticeable in case of small images, and with larger imagery, the overhead can be reduced to 10% of processing time.

Based on the experimental results, object based classification has proven to be a valid choice both in terms of accuracy and performance. For segmentation the graph-based method is suggested. The tiling of images comes with great performance benefit.

## 6 Conclusion and future work

The paradigm shift to cloud computing can be a tough challenge, and usually requires a great deal of effort, because data management has to be reconsidered, algorithms have to be redesigned.

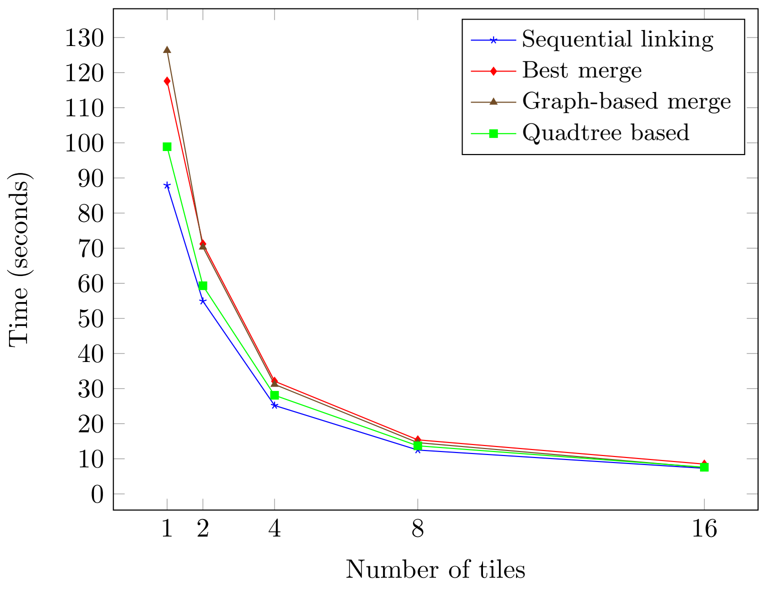


Figure 6: Average execution time on medium images

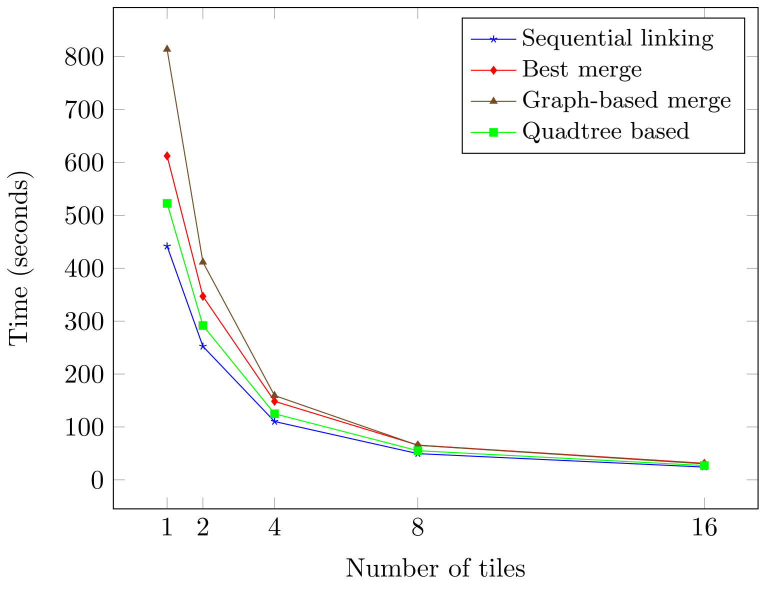


Figure 7: Average execution time on large images

In this paper, flood and waterlogging detection was presented as a case study. The algorithm is a good candidate for advancement, as the current solution does not satisfy all requirements and cannot be applied in all scenarios. The workflow was enhanced both in terms of architecture and algorithm – enabling quick automated response using unsupervised classification. The complete reimplementation was avoided, as algorithms developed for single machine environment can be applied in the distributed environment by following our methodology. Usually, the application of the methodology requires the introduction of additional post-processing operations, however, the cost of implementing these operations is far less than starting from scratch.

Experiments showed that the automated workflow provides over 90% accuracy with respect to the original algorithm, which is more than enough in most cases. The results can be computed within a matter of minutes if the cloud environment is large enough. Although this process does not involve user interaction, expert knowledge is built into the system as the reference areas are selected from the supervised classification results.

In its current form, the process is already applicable, but further study will be performed to enable more accuracy and better responsiveness. To aid in specializing the process for performing well even in any edge cases further clustering and segmentation algorithms will be added. To wider the set of usable execution environments shifting local processing to utilize GPGPU computing will be examined as well.

## References

[1] Agrawal, D., Das, S., and El Abbadi, A. Big Data and Cloud Computing: Current State and Future Opportunities. In *Proceedings of the 14th International Conference on Extending Database Technology*, EDBT/ICDT '11, pages 530–533, 2011.

[2] Aji, A., Wang, F., Vo, H., Lee, R., Liu, Q., Zhang, X., and Saltz, J. Hadoop GIS: A High Performance Spatial Data Warehousing System over Mapreduce. *Proc. VLDB Endow.*, 6(11):1009–1020, 2013.

[3] Alonso-Calvo, R., Crespo, J., Garc'ia-Remesal, M., Anguita, A., and Maojo, V. On distributing load in cloud computing: A real application for very-large image datasets. *Procedia Computer Science*, 1(1):2669–2677, 2010.

[4] Banaei, S.M. and Moghaddam, H.K. Hadoop and Its Role in Modern Image Processing. *Open Journal of Marine Science*, 4(4):239–245, 2014.

[5] Bednarz, T., Wang, D., Arzhaeva, Y., Lagerstrom, R., Vallotton, P., Burdett, N., Khassapov, A., Szul, P., Chen, S., Sun, C., Domanski, L., Thompson, D., Gureyev, T., and Taylor, J.A. Cloud Based Toolbox for Image Analysis, Processing and Reconstruction Tasks. In *Signal and Image Analysis for Biomed-*

*ical and Life Sciences*, volume 823 of *Advances in Experimental Medicine and Biology*, pages 191–205. 2015.

[6] Bhandarkar, M. MapReduce programming with Apache Hadoop. In *Parallel Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, pages 1–1, 2010.

[7] Blaschke, T. Object based image analysis for remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(1):2–16, 2010.

[8] Brakenridge, G. R., Andersona, E., Nghiemb, S. V., Caquard, S., and Shabaneh, T. B. Flood Warnings, Flood Disaster Assessments, and Flood Hazard Reduction: The Roles of Orbital Remote Sensing. In *Proceedings of the 30th International Symposium on Remote Sensing of Environment*, pages 1–6, 2003.

[9] Chen, S., Bednarz, T., Szul, P., Wang, D., Arzhaeva, Y., Burdett, N., Khassapov, A., Zic, J., Nepal, S., Gurevey, T., and Taylor, J. Galaxy + Hadoop: Toward a Collaborative and Scalable Image Processing Toolbox in Cloud. In *Service-Oriented Computing ICSOC 2013 Workshops*, volume 8377 of *Lecture Notes in Computer Science*, pages 339–351. 2014.

[10] Choi, E. and Lee, C. Feature extraction based on the Bhattacharyya distance. *Pattern Recognition*, 36(8):1703–1709, 2003.

[11] Codella, N.C.F., Hua, G., Natsev, A., and Smith, J.R. Towards large scale land-cover recognition of satellite images. In *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on*, pages 1–5, 2011.

[12] Csornai, G. Disaster Monitoring With The Integrated Utilization Of Envisat and Other Satellite Data Sets in the 2004-2006 Period In Hungary. In *Proceedings of the Envisat Symposium*, 2007.

[13] Csornai, G., Mikus, G., Nádor, G., Hubik, I., László, I., and Suba, Z. The first seven years of the remote sensing based Ragweed Monitoring and Control System. In *EARSeL eProceesings*, pages 110–118, 2011.

[14] de Oliveira, D., Baião, F.A., and Mattoso, M. Towards a Taxonomy for Cloud Computing from an e-Science Perspective. In Antonopoulos, Nick and Gillam, Lee, editors, *Cloud Computing*, Computer Communications and Networks, pages 47–62. 2010.

[15] Dean, J. and Ghemawat, S. MapReduce: Simplified Data Processing on Large Clusters. In *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation*, volume 6 of *OSDI'04*, pages 10–10, 2004.

[16] Dezső, B., Fekete, I., Gera, D., Giachetta, R., and László, I. Object-based image analysis in remote sensing applications using various segmentation techniques. *Ann. Univ. Sci. Budap. Rolando Eötvös, Sect. Comput.*, 37:103–120, 2012.

[17] Dezső, B., Giachetta, R., László, I., and Fekete, I. Experimental Study on Graph-based Image Segmentation Methods in the Classification of Satellite Images. In *EARSel eProceedings*, volume 11, pages 12–14, 2012.

[18] El Hajj, M., Bégué, A., Lafrance, B., Hagolle, O., Dedieu, G., and Rumeau, M. Relative Radiometric Normalization and Atmospheric Correction of a SPOT 5 Time Series. *Sensors*, 8(4):2774–2791, 2008.

[19] Eldawy, A. and Mokbel, M.F. A Demonstration of SpatialHadoop: An Efficient Mapreduce Framework for Spatial Data. *Proc. VLDB Endow.*, 6(12):1230–1233, 2013.

[20] Felzenszwalb, P.F. and Huttenlocher, D.P. Efficient Graph-Based Image Segmentation. *Int. J. Comput. Vision*, 59(2):167–181, 2004.

[21] Giachetta, R. AEGIS - A state-of-the-art spatio-temporal framework for education and research. *OSGeo Journal*, 13(1):68–77, 2014.

[22] Giachetta, R. A framework for processing large scale geospatial and remote sensing data in MapReduce environment. *Computers & Graphics*, 2015.

[23] Giloi, W. Distributed Image Processing. In *Advances in Digital Image Processing*, pages 249–263. 1979.

[24] Golpayegani, N. and Halem, M. Cloud Computing for Satellite Data Processing on High End Compute Clusters. In *Cloud Computing, 2009. CLOUD '09. IEEE International Conference on*, pages 88–92, 2009.

[25] Gray, J., Liu, D.T., Nieto-Santisteban, M., Szalay, A., DeWitt, D.J., and Heber, G. Scientific Data Management in the Coming Decade. *SIGMOD Rec.*, 34(4):34–41, 2005.

[26] Gueld, M.O., Thies, C.J., Fischer, B., Keysers, D., Wein, B.B., and Lehmann, T.M. Platform for distributed image processing and image retrieval. volume 5150, pages 1109–1120, 2003.

[27] Hare, J.S., Samangooei, S., and Lewis, P.H. Practical scalable image analysis and indexing using Hadoop. *Multimedia Tools and Applications*, 71(3):1215–1248, 2014.

[28] Herring, J.R., editor. *OpenGIS Implementation Standard for Geographic Information: Simple Feature Access - Common Architecture, version 1.2.1.* Open Geospatial Consortium, 2011.

[29] Jinnan, Y. and Sheng, W. Studies on application of cloud computing techniques in GIS. In *Geoscience and Remote Sensing (IITA-GRS), 2010 Second IITA International Conference on*, volume 1, pages 492–495, 2010.

[30] Karimi, H.A., editor. *Big data: techniques and technologies in geoinformatics.* CRC Press, 2014.

[31] Kocakulak, H. and Temizel, T.T. A Hadoop solution for ballistic image analysis and recognition. In *High Performance Computing and Simulation (HPCS), 2011 International Conference on*, pages 836–842, 2011.

[32] Krämer, M. and Senner, I. A modular software architecture for processing of big geospatial data in the cloud. *Computers & Graphics*, 2015.

[33] Kriegler, F.J., Malila, W.A., Nalepka, R.F., and Richardson, W. Preprocessing transformations and their effects on multispectral recognition. In *Remote Sensing of Environment, VI*, volume 1, page 97, 1969.

[34] László, I. The integration of remote sensing and GIS data in the control of agricultural subsidies in Hungary. In *Proceedings of the 33rd Symposium of EARSeL*, pages 589–598, 2013.

[35] László, I., Ocsai, K., Gera, D., Giachetta, R., and Fekete, I. Object-based image analysis of pasture with trees and red mud spill. In *Proceedings of the 31th EARSeL Symposium*, volume 13, pages 423–431, 2011.

[36] Lee, C.A., Gasster, S.D., Plaza, A., Chang, C., and Huang, B. Recent Developments in High Performance Computing for Remote Sensing: A Review. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 4(3):508–527, 2011.

[37] Lee, K., Lee, Y., Choi, H., Chung, Y.D., and Moon, B. Parallel Data Processing with MapReduce: A Survey. *SIGMOD Rec.*, 40(4):11–20, 2012.

[38] Li, B., Zhao, H., and Lv, Z.H. Parallel ISODATA Clustering of Remote Sensing Images Based on MapReduce. In *Proceedings of the 2010 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, CYBERC '10, pages 380–383, 2010.

[39] Li, X.L., Veeravalli, B., and Ko, C.C. Distributed image processing on a network of workstations. *International Journal of Computers and Applications*, 25(2):1–10, 2003.

[40] Lin, F., Chung, L., Ku, W., Chu, L., and Chou, T. The Framework of Cloud Computing Platform for Massive Remote Sensing Images. In *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*, pages 621–628, 2013.

[41] Madhukar, B.N. and Narendra, R. Lanczos Resampling for the Digital Processing of Remotely Sensed Images. In *Proceedings of International Conference on VLSI, Communication, Advanced Devices, Signals & Systems and Networking (VCASAN-2013)*, volume 258 of *Lecture Notes in Electrical Engineering*, pages 403–411. 2013.

[42] Malakar, R. and Vydyanathan, N. A CUDA-enabled Hadoop cluster for fast distributed image processing. In *Parallel Computing Technologies (PARCOMPTECH), 2013 National Conference on*, pages 1–5, 2013.

[43] Merritt, A.M., Gupta, V., Verma, A., Gavrilovska, A., and Schwan, K. Shadowfax: Scaling in Heterogeneous Cluster Systems via GPGPU Assemblies. In *Proceedings of the 5th International Workshop on Virtualization Technologies in Distributed Computing*, VTDC '11, pages 3–10, 2011.

[44] Moise, D., Shestakov, D., Gudmundsson, G., and Amsaleg, L. Indexing and Searching 100M Images with Map-Reduce. In *Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval*, ICMR '13, pages 17–24, 2013.

[45] Shvachko, K., Kuang, H., Radia, S., and Chansler, R. The Hadoop Distributed File System. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10, 2010.

[46] Tan, Y.S., Tan, J., Chng, E.S., Lee, B., Li, J., Date, S., Chak, H.P., Xiao, X., and Narishige, A. Hadoop framework: impact of data organization on performance. *Software: Practice and Experience*, 43(11):1241–1260, 2013.

[47] Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Zhang, N., Antony, S., Liu, H., and Murthy, R. Hive – a petabyte scale data warehouse using Hadoop. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*, pages 996–1005, 2010.

[48] Wang, P., Wang, J., Chen, Y., and Ni, G. Rapid processing of remote sensing images based on cloud computing. *Future Generation Computer Systems*, 29(8):1963–1968, 2013.

[49] Yan, Y. and Huang, L. Large-Scale Image Processing Research Cloud. In *Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING), 2014 International Conference on*, pages 88–93, 2014.

[50] Yang, C., Goodchild, M., Huang, Q., Nebert, D., Raskin, R., Xu, Y., Bambacus, M., and Fay, D. Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? *International Journal of Digital Earth*, 4(4):305–329, 2011.

[51] Yang, Z., Kamata, S., and Ahrary, A. NIR: Content based image retrieval on cloud computing. In *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, volume 3, pages 556–559, 2009.

[52] Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., and Stoica, I. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, pages 10–16, 2010.

[53] Zhang, C., Sterck, H., Aboulnaga, A., Djambazian, H., and Sladek, R. Case Study of Scientific Data Processing on a Cloud Using Hadoop. In *High Performance Computing Systems and Applications*, volume 5976 of *Lecture Notes in Computer Science*, pages 400–415. 2010.