TARTU UNIVERSITY
Faculty of Mathematics and Computer Science
Institute of Computer Science
Computer Science

Karl Potisepp

# Large-scale image processing using MapReduce

M.Sc. Thesis (30 ECTS)

Supervisor: Pelle Jakovits M.Sc., Satish Narayana Srirama Ph.D.

Author: . . . . . " . . . . . " april 2013

Supervisor: . . . . . " . . . . . " april 2013

Approved for defence

Professor: . . . . . " . . . . . " april 2013

Tartu 2013

# Contents

# Chapter 1

# Introduction

Along with the development of information technology, a constant stream of new applications for solving humanity's problems has also appeared. As we possess more computing power, we can tackle more and more resource-intensive problems such as DNA sequencing, seismic imaging and weather simulations. When looking at these subjects, a common theme emerges: all of these involve either analysis or generation of large amounts of data. While personal computers have gone through a staggering increase in power during the last 20 years, and the processing power even within everyday accessories - such as smartphones - is very capable of solving problems that were unfeasible for supercomputers only a couple of decades ago, analysing the amount of data generated by newest generation scientific equipment is still out of reach in some areas. Moreover, as processor architectures are reaching their physical limitations (TODO explain this in more detail?), using distributed computing technologies has become a popular way to solve problems which do not fit the confines of a single computer. Supercomputers, GRID-based systems and computing clouds are an example of this approach. Since the fields of distributed computing and image processing are too broad to fully cover in this thesis, this work will focus on the latter of the three with regard to image processing.

Image processing, high resolution medical images, cheap quality cameras and abundance of photos. Also cheap hard drives which allow to store all this stuff. TODO explain :)

In this thesis, I will describe solving two different image processing tasks related to two-dimensional colour images using the Hadoop open source distributed computing framework.

Description in broad terms of what has been achieved with this thesis.

Possible subtopics:

- Microscope photo processing with bilateral filter - Datexim SAS (ENSICAEN)
- de-noising satellite images (maybe)
- Classification and metadata extraction from archaeological excavation data (TO BE ELABORATED)- Graeme Earl and Hembo Pagi (U of Southampton)

## 1.1 Problem statement

Processing of large images can not be done on a single personal computer, because they do not fit into memory. Processing of many small images also requires distributed computing because it takes too much time. Neither of these issues will reliably be solved by advances in computing power, because CPUs are already reaching their theoretical physical limitations and the scale of data is increasing faster than the processing capabilities of a single computer.

A solution for these problems is turning towards distributed computing, where limitations of a single computer are dealt with by combining the resources of many computers to perform one large task. While this approach is not new - supercomputers and computing clusters have existed for many years already - it is only recently that techniques of using commodity computers for distributed processing have gained popularity. Combining this with the ability to rent virtual machines from cloud providers such as Amazon and Microsoft it is easy to see how solving large scale processing tasks is now feasible for those who do not have access to a supercomputer nor the resources to build or maintain one.

### 1.1.1 Image processing

Lots of images on the internet, youtube, produced by high definition cameras. Realtime processing and feature extraction, denoising. Focus on 2-dimensional color images, subfocus on one large image and a dataset of many small images with regard to MapReduce.

### 1.1.2 Why MapReduce?

Problem statement. What is the problem, why does it need to be solved, why solve it this way?

What are the alternatives? batch processing on the PC is feasible for only small amounts of data; maintaining a purpose-built computer cluster is only cost-effective for very large and important problems; Hadoop is affordable and easy to use, yet can be used even if the data scales to huge sizes making it ideal for this (however we need to find out whether that is the case)

# Chapter 2

# Background

## 2.1 Relevant work

Information on how others have solved similar problems (if any), and what are the differences in approach and why the methodology described in this work is different enough to warrant attention.

- Web-Scale Computer Vision using MapReduce for Multimedia Data Mining [11]: A case study of classifying and clustering billions of regular images using MapReduce. No mention is made of average image dimensions or any issues with not being able to process certain images because of memory limitations. However, a way of pre-processing images for use in a sliding-window approach for object recognition is described. Therefore one can assume that in this approach, the size of images is not an issue, because the pre-processing phase cuts everything into a manageable size. The question still remains whether a sliding window approach is capable of recognizing any objects present in the image that do not easily fit into one analysis window, and whether the resource requirements for image classification and image processing are significantly different or not.

- An Architecture for Distributed High Performance Video Processing in the Cloud [8]: This article outlines some of the limitations of the MapReduce model when dealing with high-speed video encoding, namely it's dependence on the NameNode as a single point of failure (however a fix is claimed at [3]), and lack of possibility for generalization in order to suit the issue at hand. An alternative - optimized - implementation is proposed for providing a cloud-based IaaS solution. However, considering the advances of distributed computation technology within the past two years (the article was published in 2010) and

the fact that the processing of truly large images was not touched upon, the problem posed in this work still remains.

- Clustering Billions of Images with Large Scale Nearest Neighbor Search [1]: Here, a MapReduce-based approach for parallelizing nearest-neighbor clustering is described. The report focuses more on the technicalities of adapting a spill-tree based approach for use on multiple machines. Also, a way for compressing image information into smaller feature vectors is described. With regards to this thesis, again the focus is not so much on processing the images to attain some other result than something intermediate to be used in search and clustering.

- Parallel K-Means Clustering of Remote Sensing Images Based on MapReduce [7]: A description of sing the k-means algorithm in conjunction with MapReduce and satellite/aerophoto images in order to find different elements based on their color (i.e. separate trees from buildings). Not much is told about encountering and overcoming the issues of analyzing large images besides mentioning that a non-parallel approach was unable to process images larger than 1000x1000 pixels, and that the use of a MapReduce-based parallel processor required the conversion of TIFF files into a plaintext format.

- Case Study of Scientific Data Processing on a Cloud Using Hadoop [12]: This article describes the methods used for processing sequences of microscope images of live cells. The images and data units in question are relatively small - 512x512 16-bit pixels, stored in folders measuring 90MB - there were no issues with regard to having to split data for fitting into Hadoop DFS chunks besides improving upon a couple of Java classes. No mention was made about the algorithm used to extract data from the images besides that it was written in MATLAB, which meant that the processing chain had much more complexity than just a MapReduce job.

- Using Transaction Based Parallel Computing to Solve Image Processing and Computational Physics Problems [5]: This article describes the use of distributed computing with two examples - video processing/analysis and subsurface transport. The main focus is put on the specifications of the technology used (Apache Hadoop, PNNL MeDICI), whereas there is no information presented on how the image processing parts of the examples given were implemented.

- Distributed frameworks and parallel algorithms for processing large-scale geographic data [6]: Here, many issues with processing large sets

of geographic information systems (commonly known as GIS) data are described and solved in order to be able to extract useful knowledge from it. This article was published in 2003, so while some of the issues have disappeared due to the increase in computing power available to scientists, problems stemming from the ever-increasing amount of data generated by different types of monitoring technologies (such as ensuring distribution of data to computation nodes and storing big chunks of data in memory) still remain. Also, considering that the Amazon EC2 [4] web service came online just in 2006, it is obvious that one can not make an apt comparison whether or not a MapReduce-based solution in 2012 is better or not for large-scale image processing than what was possible using grid technology in 2003.

- A Scalable Image Processing Framework for gigapixel Mars and other celestial body images [9]: This article describes the way NASA handles processing of celestial images captured by the Mars orbiter and rovers. Clear and concise descriptions are provided for the segmentation of gigapixel images into tiles, how these tiles are processed, and how the image processing framework handles scaling and works with distributed processing. The authors used the Kakadu JPEG2000 encoder and decoder along with the Kakadu Java Native Interface to develop their own processing suite. The software is proprietary and requires the purchase of a license to use.

- Ultra-fast processing of gigapixel Tissue MicroArray images using high performance computing [10]: This article talks about speeding up the analysis of Tissue MicroArray images by substituting human expert analysis for automated processing algorithms. While the images sizes processed were measured in gigapixels, the content of the image (scans of tissue microarrays) was easily segmented and there was no need to focus on being able to analyse all of the image at once. Furthermore, the work was all done on a specially built grid high performance computing platform with shared memory and storage, whereas this thesis is focused on performing processing on a MapReduce cluster.

In this chapter I will provide a thorough description of all technologies and methodologies that are important with regards to this work. I will describe the operating philosophy of the MapReduce model and the capabilities and limitations of it's most popular freely non-proprietary implementation - Hadoop. Also, the different image processing and Java-C++ technologies used while working on this thesis, such as the CImg library and the Java Native Interface, will be elaborated.

## 2.2 MapReduce

MapReduce is a programming model developed by Google for processing and generating large datasets used in practice for many real-world tasks [2]. In this section, I will focus on describing the general philosophy and methodology behind this model, whereas the following part will describe in more detail one of the more popular implementations of the model - Hadoop - which is also used for all the practical applications featured in this work.

The basic idea behind MapReduce is based on the observation that a lot of processing tasks involving large amounts of data (measured in terabytes) all need to deal with the issues of distributing the data across a network of computers to ensure that the available storage and CPU resources are maximally utilised, and it would be easier if programmers could focus on writing the actual processing part that differs task by task.

Description of the general philosophy of MapReduce, the logic behind Map/Reduce/etc. parts of the processing chain, and how this approach suits/is cumbersome when dealing with image processing.

### 2.2.1 Apache Hadoop

Hadoop is an open-source framework for distributed computing developed by the Apache Foundation, inspired by Google's MapReduce (TODO cite Hadoop and MapReduce papers). It has been in development since 2005 and - at the time of writing this work - is one of the most popular freely available applications of it's kind. Due to this and also a large user base ranging from home users to large companies makes Hadoop a good candidate for attempting to use it for image processing.

#### Hadoop Distributed File System (HDFS)

HDFS is the underlying file system that Hadoop uses in order to store data meant to be processed by MapReduce. It is inspired by the Google File System (GFS) (TODO cite). There are two reasons why the file system comes into play when discussing image processing using Hadoop. Firstly the distributed nature of HDFS ensures that all input data is spread across the cluster's storage so that at least initially every worker can process data stored on it's local hard drive. Secondly, the configured block size somewhat restricts the storage of files - if a file is larger than the block size and can not easily be split, then all it's blocks will have to be stored in the same location, which in essence means that the advantages from having a small block size are lost.

## 2.3 Image processing

Description of what is meant by image processing within the scope of this thesis.

### 2.3.1 ImageJ

Information about the Imagej library.

## 2.4 Image processing and MapReduce

Description of different ways the MapReduce method has been used in conjunction with image processing libraries. Also some use cases.

# Chapter 3

# Details of methodology

Descriptions on what was implemented, how it was implemented, problems encountered, and what has been done to solve these problems. If there were unsolvable problems, then how do these limit the applicability of the methodology in the real world and how to work around these limitations.

# Chapter 4

# Use case

Description of how the methodology established in this thesis was applied to real-world problems.

## 4.1 Analysis and processing of a large image

### 4.1.1 Description of the data and use case

What sort of image, why does it need processing?

### 4.1.2 Bilateral Filter Algorithm

In order to somehow gauge the feasibility of using Hadoop MapReduce as a platform for image processing, I used the implementation of a bilateral filter by Chaudhury et al. (TODO cite).

#### Naive bilateral filter algorithm

Description of the bilateral filter algorithm, it's applications and details about the implementation used for benchmarking.

### 4.1.3 Performance

One of the main issues with regard to measuring the performance of the FBF algorithm was comparing the results of the sequential version to the distributed MapReduce one.

## 4.2 Analysis of many small images

### 4.2.1 Description of the data and possible use case

The data set consists of 40326 JPEG encoded images (totaling at about 308GB) taken across the span of 9 years at the Portus archaeological excavation site near Rome, Italy. It is somewhat structured and some images are also tagged with meta-data using folder structures. However the structuring is incomplete, evidenced by duplicates and missing meta-data (which currently has to be manually added). The motivation behind applying distributed image processing on this data set is to find out whether it is possible to speed up the structuring process and extract some meta-data with minimal human interference.

The downside of using this data set to illustrate the analysis of many small images using MapReduce is that since there is no ideally structured version of it to compare to, all efficiency statistics will be purely subjective. However the intent of this work is not to describe an effective meta-data extraction method, but rather analyse the suitability of MapReduce for this sort of task and discuss the issues (both solved and unsolved) encountered.

### 4.2.2 Problem statement

Extracting meaningful data from images is a non-trivial task. OCR, object recognition, identifying similar images, measuring image quality.

Problem - how to feed images into MapReduce when the folder structure is complex? Hadoop does not provide a way for recursively going through the input folder and starts a new map task for every folder which creates unnecessary overhead. Creating a large SequenceFile to store all file data along with original full paths is not an option because it is too resource-intensive and the resulting file may be difficult to store. Flattening the structure and writing original paths into image meta-data also requires too much processing (reading and writing all files in the data set). Ignoring the path info altogether is not an option, because it provides a structuring/grouping of the data not encoded into meta-data.

Problem - since images come from many different makes and models of cameras, and there is no easy way in distinguishing between meta-data that is useful in the context of this task and meta-data that encodes some cryptic information that is only used by some proprietary software (for example serial numbers and firmware version numbers) or simply the parameters of the camera at the time of taking the photo (such as Base ISO, Auto Focus settings etc.). While some of the latter meta-data may be useful when searching for

similar images (for instance, one can group together photos taken with flash), there is no way of specifying which meta-data key refers to the correct value without first looking through a representative sampling (or even all) of the images and manually filtering out the meta-data keys to group together. (TODO maybe explain this better)

### 4.2.3 Performance

## 4.3 Overview and discussion

This thesis does blabla and provides blabla with regard to MapReduce.

# Bibliography


[1] *Clustering Billions of Images with Large Scale Nearest Neighbor Search*, 2007.

[2] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008.

[3] Borthakur Dhruba. Looking at the code behind our three uses of apache hadoop, December 2010.

[4] Borthakur Dhruba. Amazon elastic compute cloud (amazon ec2), October 2012.

[5] R. Farber S. Elbert H. Trease, D. Fraser. Using transaction based parallel computing to solve image processing and computational physics problems.

[6] Kenneth A. Hawick, P. D. Coddington, and H. A. James. Distributed frameworks and parallel algorithms for processing large-scale geographic data. *Parallel Comput.*, 29(10):1297–1333, October 2003.

[7] Zhenhua Lv, Yingjie Hu, Haidong Zhong, Jianping Wu, Bo Li, and Hui Zhao. Parallel k-means clustering of remote sensing images based on mapreduce. In *Proceedings of the 2010 international conference on Web information systems and mining*, WISM'10, pages 162–170, Berlin, Heidelberg, 2010. Springer-Verlag.

[8] Rafael Pereira, Marcello Azambuja, Karin Breitman, and Markus Endler. An architecture for distributed high performance video processing in the cloud. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, CLOUD '10, pages 482–489, Washington, DC, USA, 2010. IEEE Computer Society.

[9] M.W. Powell, R.A. Rossi, and K. Shams. A scalable image processing framework for gigapixel mars and other celestial body images. In *Aerospace Conference, 2010 IEEE*, pages 1 –11, march 2010.

[10] Y. Wang, D. McCleary, C. Wang, P. Kelly, J. James, D.A. Fennell, and P.W. Hamilton. Ultra-fast processing of gigapixel tissue microarray images using high performance computing. *Cell Oncol (Dordr)*, 34(5):495–507, 2011.

[11] Brandyn White, Tom Yeh, Jimmy Lin, and Larry Davis. Web-scale computer vision using mapreduce for multimedia data mining. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, MDMKDD '10, pages 9:1–9:10, New York, NY, USA, 2010. ACM.

[12] Chen Zhang, Hans De Sterck, Ashraf Aboulnaga, Haig Djambazian, and Rob Sladek. Case study of scientific data processing on a cloud using hadoop. In *Proceedings of the 23rd international conference on High Performance Computing Systems and Applications*, HPCS'09, pages 400–415, Berlin, Heidelberg, 2010. Springer-Verlag.