IN4392 Cloud Computing
Cloud Search Report

Ruxandra Cioromela - (student number)
Vincent Ghiëtte - 1358251

November 5, 2014

1 Cloud Search

1.1 The application

Cloud Search (CS) is an application which can be deployed on a cloud service. The application can be categorized as a search engine. It asks the user for input, person, location or organization name, and returns relevant news articles to the user.
The main difference between Cloud Search and classic search engines is that CS uses natural language processing (NLP). Using NLP Cloud Search extracts the names of persons, locations and organisations. Then it compares the found names in the articles against the given names by the user. After the comparison, the application returns the relevant texts to the user.
Additional to returning the news articles, CS also returns the names present in the article and their occurrence in that text. Returning the names and their occurrence may give the user additional insight on his search query, as names which often occur may be related to his search query.

1.2 Application structure

The application has three main parts, the web-page, master and slave part. All the parts work together in order to return relevant search results to the user.

1.2.1 Web-page

The web-page is the only visible aspect to the user. The web-page allows the user to formulate queries, and it displays the results to the user. The page runs entirely on JavaScript and is dependant on hosted library files. This translates in the ability for users to download the page and to run it on their local machines.
Furthermore, the page uses Bootstrap[1] and Jquery[2] to enhance the usability by offering a good looking page.

[1] `http://getbootstrap.com/`

[2] `http://jquery.com/`

1.2.2 Master

The master part of the program is written entirely in Java. It uses the Spark[3] to handle incomming POST requests. The master part also uses the AWS sdk[4] to communicate with the S3 storage. Furthermore, the Google Json library [5] is used to parse objects into their Json equivalent.
The master processes a query sent by the user. Then it retrieves the number of news articles stored on S3. Next, the master assigns the analysis of texts to slaves. The master decides which texts are analysed by which slave. Effectively he distributes the work amongst the slaves which can do the analysis in parallel. The results of the slaves are then merged by the master and returned to the user.

1.2.3 Slave

The slave is also written entirely in Java and uses the same libraries as the master. Additionally the slaves use the OPENNLP[6] library to analyse texts.
Upon receipt an analysis request of the master, the slave download the assigned article files form the S3 instance. The files are then analysed using OPENNLP. Next the files are filtered against the search parameters and the relevant files with the names and occurrences are returned to the master.

1.3 Additional resource

In addition to the application an elementary testing web-page is included in the project. The test page is similar to the web-page offered to the user. In addition it uses the Google charting libraries[7]
The testing page allows to test the response time of Cloud Search. I allows the user to specify the number of files each slave should be assigned to by its master. Furthermore it allows the tester to chose the amount of simultaneous search requests to send to Cloud Search.
The results are shown on a graph and the comma separated value file can be downloaded with the raw .

[3] http://sparkjava.com/

[4] http://aws.amazon.com/sdk-for-java/

[5] https://code.google.com/p/google-gson/

[6] https://opennlp.apache.org/

[7] https://developers.google.com/chart/