UNIVERSITY OF CYPRUS

Department of Computer Science

EPL 660 – Information Retrieval and Search Engines

Assignment 2 – Development of Distributed and Scalable Information Retrieval System: «DSPythia»

Tutor: George Pallis / Assignment Instructor: Pavlos Antoniou

Posting date: Monday 14/10/19

Deadline: Friday 01/11/19 @ 23:59 noon (18 days)

(solution to be submitted as zip via Moodle)
http://www.cs.ucy.ac.cy/courses/EPL660

1. Goal

The purpose of this assignment is to provide an opportunity for the student to integrate and apply various data processing and analysis tools such as ElasticSearch, Apache Hadoop (implementing the Map-Reduce model), Apache Lucene, Apache Solr, and the Vector Space model.

2. Requirements

In the first assignment you developed a system for processing, storing, searching and retrieving information called Dionysos. The system was using inverted indexes to index wine review collections and supported the Boolean model for querying the indexes.

The inverted index of Dionysos system was stored in memory and hard disk of a single machine. As the volume of input data increases, the need for distributed and scalable data processing, indexing and storage becomes of prime importance.

In this assignment, you will develop a distributed and scalable system for information processing, storage, searching and retrieval, namely **DSPythia (Distributed Scalable Pythia)** with a **full scalable and distributed inverted index**. Information searching and retrieval from the index will be based on the **Vector Space model**.

DSPythia will include the following subsystems:

- User interface subsystem
- Collection and inverted index management subsystem. The actions allowed on the inverted index **are the same with assignment 1** (e.g. create/delete collection, open/close collection, etc.)
- Search subsystem
    - Query representation based on Vector Space Model.

- Ability to search for one or more terms.
- In each search, the system returns the files containing the search term, and the time it took to perform the search.

You may use one or more of the following: ElasticSearch or/and, Apache Hadoop or/and Apache Lucene or/andApache Solr etc.

Evaluate your implementation (using appropriate metrics) and **interpret the results**. For Για την αξιολόγηση σας θα βρείτε ιδιαιτέρως χρήσιμο να χρησιμοποιήσετε επερωτήματα (queries) που δίνονται μαζί με τη συλλογή δεδομένων (dataset) που παρουσιάζεται πιο κάτω (παράγραφος 5) και να συγκρίνετε τα αποτελέσματα που θα λάβετε από το σύστημά σας μαζί με τα αποτελέσματα της αξιολόγησης που περιέχεται στο dataset.

3. Deliverables

You must deliver via Moodle a single ZIP file including:

- Source code with comments – if you implement the assignment in Eclipse, please submit the whole eclipse project,
- Documentation (in a separate .docx) presenting your way of thinking when developing the system, listing the different classes you developed along with their operation as well as the algorithms and techniques used. The documentation must include instructions on how to run and use the application.

4. Dataset

The collection of documents that will be used for evaluating DSPythia is Cranfield. This collection contains 1399 abstracts from articles that were published inaerodynamics journals in late 50s. It is a relatively small scale collection which is largely used in the field of IR for pilot experimental application such as the current assignment. The collection is available in: http://www.cs.ucy.ac.cy/courses/EPL660/exercises/cran.tar.gz and contains the following files:

- cran.all – All articles in a single text file. For each article the following features are given: I (index), T (title), A (author), B (bibliography) and W (words). The latter refers to the abstract of each article.
- cran.qry – A set of 225 queries. Each query is characterized by its I (index) and the number of the terms W (words) of the query.
- cranqrel – Evaluation of the relevance of queries to articles for all pairs (query, article). In particular, each line of this file contains 3 integers corresponding to the query number, the article number, and relevance score (1-5). Explanation of the scores is given in the file below.
- cranqrel.readme – Explanation of the evaluation file.

5. References

- Apache Hadoop. http://hadoop.apache.org/
- Apache Lucene. http://lucene.apache.org/

- Apache Solr – Distributed Search. http://wiki.apache.org/solr/DistributedSearch
- Apache Solr – Running Solr on HDFS. https://cwiki.apache.org/confluence/display/solr/Running+Solr+on+HDFS
- Apache Solr – MapReduceIndexer Tool [link1] [link2].
- Solrj – Java Client to access Solr. https://wiki.apache.org/solr/Solrj, http://www.solrtutorial.com/solrj-tutorial.html
- ElasticSearch - https://www.elastic.co/
- ElastixSearch – Java API
- R. McCreadie , C. Macdonald, I. Ounis. MapReduce indexing strategies: Studying scalability and efficiency, Information Processing and Management 48 (2012) 873-888.
- D. Jeff and G. Sanjay. MapReduce: Simplified Data Processing on Large Clusters. In Sixth Symposium on Operating System Design and Implementation (OSDI'04), San Fransisco, CA, December 2004.
- D. Jeff and G. Sanjay. MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM, 51(1), 2008.
- J. Zobel. Inverted Files for Text Search Engines. ACM Computing Surveys, 38(2), 2006.