**Comparative study of information Retrieval Systems using Hadoop Map/Reduce**

Sneha Kadam Emmanuel John Soham Ashok Parekh

Department of Computer Science

Clemson University

Clemson, SC 29631

{skadam, emmanuj, sohamap}@clemson.edu

*Abstract*— With the upsurge of raw data available, and the demand for useful information within this raw data that are useful to making important business decisions, it has become increasingly important to have effective and fast information retrieval systems that will enable users to find the information they need. In this paper, we evaluate three information retrieval model – namely the standard Boolean model, the vector space model, and the probabilistic relevance model on which is based the okapi (BM25) relevance function. We also evaluate all three models using Hadoop Map/Reduce implementation and consider different methods for improving performance of the implementations.

*Index Terms*—Information retrieval, Hadoop, map/reduce.

# Introduction

Information retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources [1]. So the problem in Information retrieval is that, given a query, we need to find all the documents in our data set, which contain those words and are relevant to the context of the query. In order to solve this problem of Information Retrieval we need to follow certain steps. Firstly we need to build an index using all the files in the data set. This index contains terms in all the documents along with their document IDs in which they appear and positions in which they appear in those documents. Then we need to apply any of the ranking schemes to this index, to get all the documents relevant to the query. Thus, in this paper we explain the three ranking schemes, Boolean Retrieval, Ranked Retrieval and Okapi BM-25, which we have used to retrieve relevant documents from the data set. We have implemented all these ranking schemes using the Hadoop Map Reduce framework on a cluster and used 8 cores for executing the jobs to speed up the Information retrieval process. The dataset we have used is Reuter’s news data, which is widely used test collection for text categorization research and is collected and labeled by Carnegie Group, Inc. and Reuters, Ltd. It contains over 800k Documents in xml format. So we converted that xml content to normal text using a java parser and then used it for our research.

# Theoretical Background

1. ***Index File Construction:***

We have used Inverted Index for Boolean Retrieval System and Positional Index for the Ranked and BM-25 retrieval Systems.

The Inverted Index file contains all the terms in the dataset and the doc IDs in which they are present. The format of the Inverted Index file is as follows:

Term: docID1, docID2, docID3;

The positional index file was used in order to make it easier and quicker to retrieve document frequency of each term and their corresponding term frequency in each document. Our positional index is structured in the following way:

Term: doc\_freq; docId:pos1, pos2,pos3, pos5;docId:pos1, pos2,pos3…;

doc\_freq :- No. of documents in which the term appears.

pos1, pos2…. :- positions of occurrence in the document.

Thus, term frequency is simply the count of the positions.

1. ***Ranking Schemes:***

***Boolean Retrieval***

In order to use Boolean Retrieval scheme we need to construct Inverted Index. The Boolean Retrieval system takes as input the Inverted Index of the entire dataset and the query. It then extracts only those terms from the index which match the query terms and combines its postings list. It performs Boolean Operations like AND , OR and NOT on the postings list of the terms and result of these boolean operation is a set of documents containing all or some of the query terms, depending on the query. For example, Consider the following query:

*Brutus AND Calpurnia*

And the Inverted Index listing for Brutus and Calpurnia is as follows:

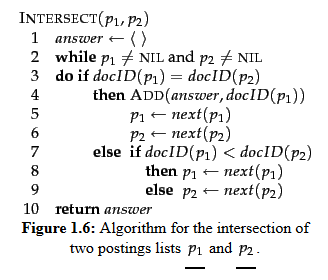
Brutus-→1→2→4→11→31→45→173→174

Calpurnia-→2→31→54→101

Here the inverted index contains all the terms in the dataset and the doc IDs in which they are present.

Then the Boolean Retrieval System applies the AND operation to the 2 listings and the result of the operation is the list of documents containing Brutus and Calpurnia.

The algorithm for the AND or Intersection operation is as follows:

******

***Ranked Retrieval:***

*Term Frequency Calculation (tf)*

The Ranked Retrieval system first calculates the term frequency which is the no. of occurrences of a word.

*Inverse Document Frequency calculation(idf)*

The idea behind calculating idf is that some have little or no discriminating power in determining relevance. For Example, a collection of documents on the auto industry is likely to have the term *auto* in almost every document. Thus idf helps in pruning out such words, which occur too often in the collection to be meaningful in relevance determination [2]. Thus formula for idf calculation is as follows:

Idft=log10 (N/dft )

Where is N no. of documents in the entire data set and dft  is the no. of documents in which the term occurs

Weight calculation for terms in Positional Index (Wt,d) :

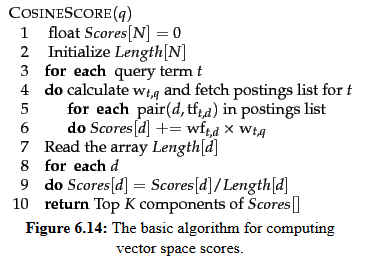
The system then multiplies tf and idf for every term in the positional index. Those are the weights assigned to the terms in the entire data set.

Weight calculation for Query terms(Wt,q):

The tf-idf weights are calculated for all the query terms. Here tf for a query term is no. of times it occurs in the query and idf is the idf calculated for the for terms in the Positional Index.

* + 1. Rank Calculation :

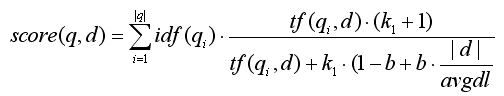
After the weights (Wt,d  and Wt,q ) have been calculated following algorithm is used to calculate the ranks for all the documents containing the query terms.



This algorithm not only calculates ranks but also sorts all the relevant documents depending on the rank and returns the top K documents.

**Okapi BM-25 Retrieval**:

This system inherits the tf and idf calculation concept from Ranked Retrieval System but uses them differently to calculate scores for the determining the relevant documents. The formula for calculation of scores is as follows:



Here,

1. tf(qi,d) correlates to the term's frequency, defined as the number of times query term qi appears in the document d .
2. |d| is the length of the document d in words (terms).
3. avgdl is the average document length over all the documents of the collection.
4. k1 and b are free parameters, usually chosen as k1 = 2.0 and b = 0.75.

Thus, after computing the scores for all the relevant documents, the system returns the top k results.

# System Design

We implemented the systems using Hadoop map/reduce. Map/Reduce is especially appropriate given that its architecture suits our needs.

* 1. Hadoop Map/Reduce

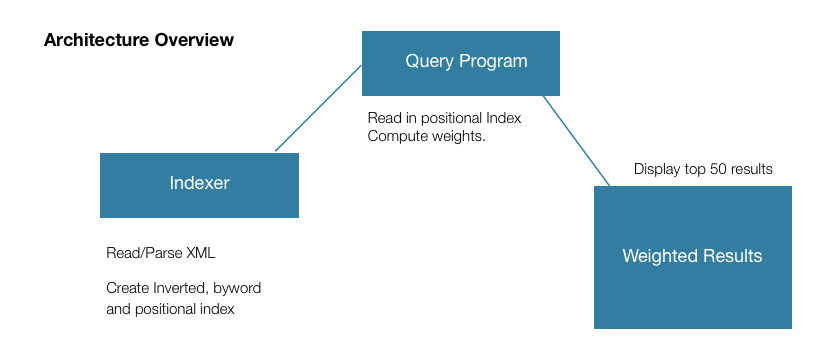
Using Map/Reduce, the data and programs can be easily distributed and shared between different nodes thereby maximize the computational resources available while running multiple tasks in parallel.

B. Helper Programs

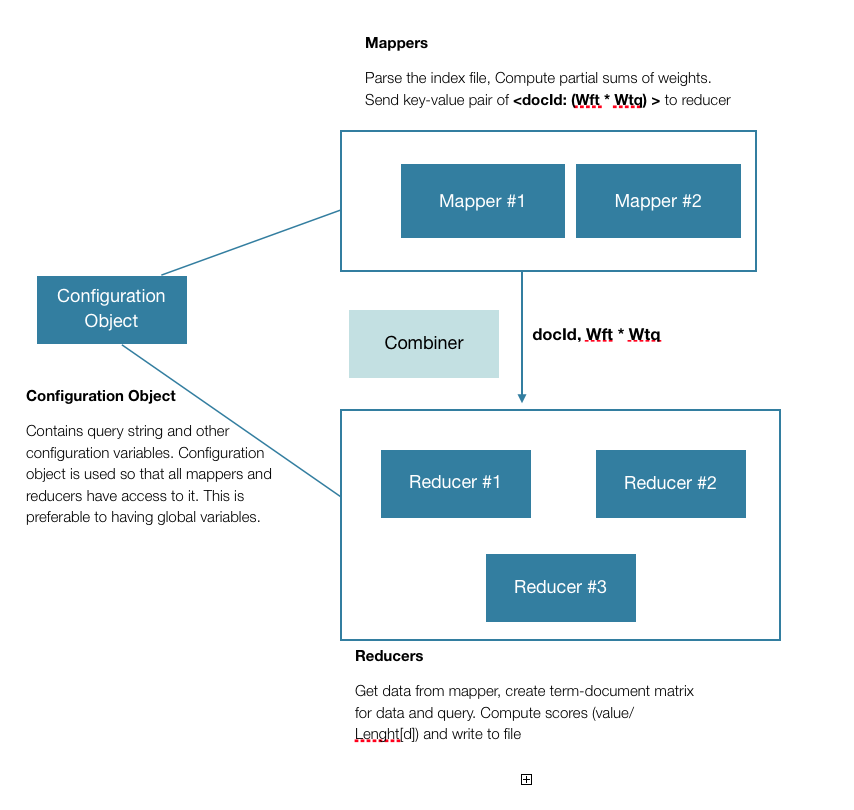
We wrote some helper programs to help compute:

1. Number of documents in the collection.
2. A key value list of the document id and the corresponding document length.

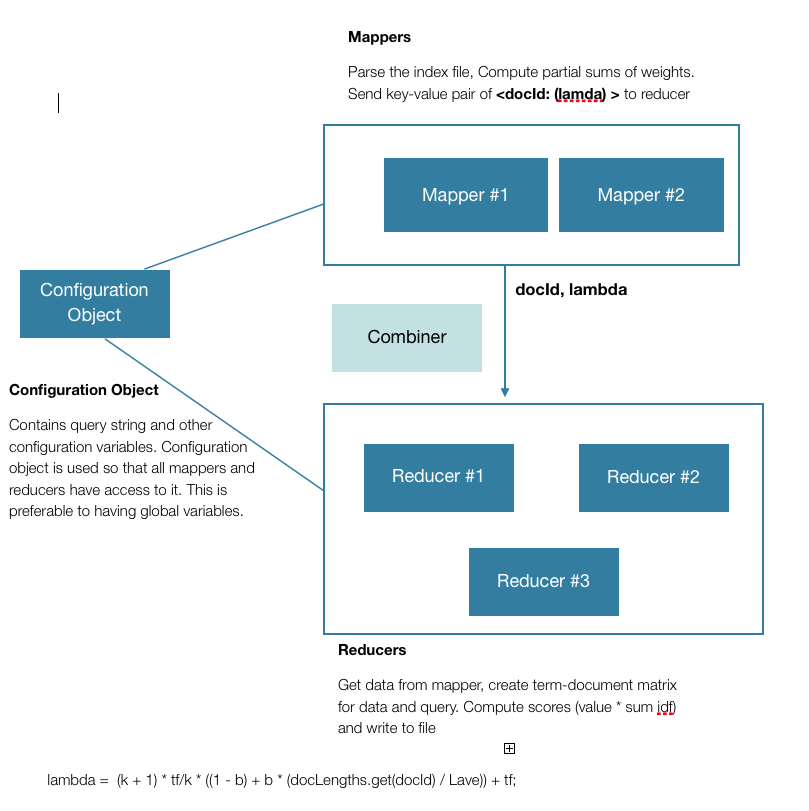
C. Architecture Overview



D. Ranked Retrieval Implementation



E. BM25 Implementation



# Experimental Results

Evaluation is very important in order to better search engines. It is also very important to understand if a search engine is being used in effectively in a specific application. Engineers don’t make decisions about a new design for a commercial aircraft based on whether it feels better than another design. Instead, they test the performance of the design by simulations and experiments, evaluate everything again when a prototype is built, and then continue to monitor and tune the performance of the aircraft after it goes into service. Experience has shown us that ideas that we intuitively feel must improve search quality, or models that have appealing formal properties, can often have little or no impact when tested using quantitative experiments.

One of the important distinctions made in the measuring how good is the search engine is between effectiveness and efficiency. Effectiveness primarily measures the ability of the search engine to gather the correct information, and efficiency is the measure of how quickly it is able to achieve this task. For a query, and a specific relevance standard, we can more precisely estimate effectiveness as a measure of how well the ranking produced by the search engine corresponds to a ranking based on user relevance judgments. Efficiency is defined in terms of the time and space requirements for the algorithm that produces the ranking. Viewed more generally, however, search is an interactive process involving different types of users with different information problems. In this environment, effectiveness and efficiency will be affected by many factors such as the interface used to display search results and techniques such as query suggestion and relevance feedback. Carrying out this type of holistic evaluation of effectiveness and efficiency, while important, is very difficult be- 2 8 Evaluating Search Engines cause of the many factors that must be controlled. For this reason, evaluation is more typically done in tightly defined experimental settings and this is the type of evaluation we focus on here.

# Theoretical Background

We are using 3 techniques as measure of the effectiveness and measure of efficiency of the search engines. The first technique used is called Precision. Precision essentially is the measure of number of relevant documents out of the total number of documents retrieved.

\begin{displaymath}
\mbox{Precision} = \frac{\char93 (\mbox{relevant items retri...
...x{retrieved items})}
= P(\mbox{relevant}\vert\mbox{retrieved})
\end{displaymath}

The second method used for evaluation is finding out the Recall. Recall is basically the number of relevant documents retrieved out of the total number relevant documents.

\begin{displaymath}
\mbox{Recall} = \frac{\char93 (\mbox{relevant items retrieve...
...x{relevant items})} =
P(\mbox{retrieved}\vert\mbox{relevant})
\end{displaymath}

The last method used for evaluation purposes is called F-measure. Once we have the precision and recall for a set of queries, the F-measure can be found out by taking the harmonic mean of Precision and recall.

\begin{example}
\begin{tabular}[t]{\vert l\vert l\vert l\vert}
\hline
& Releva...
... false negatives (fn) & true negatives (tn) \\ \hline
\end{tabular}\end{example}

Thus, Precision P = (tp/tp+fp)

Recall R = (tp/tp+tn)

The Boolean retrieval systems work on the logic of matching a particular query with its presence in a document. Thus the criteria for evaluation of the system is just to check whether or not the term or terms are present in the document. And therefore the Precision and Recall for a Boolean System will always be greater than systems like BM25 and Ranked Retrieval and will always be closer to or equal to one.

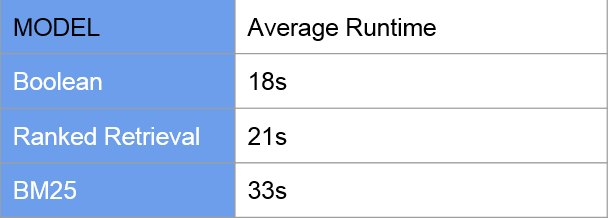
# Designing Cases for evaluation

For evaluation purposes, we used the same set of queries for evaluating BM25 and cosine similarity. After firing the queries we measured precision and recall for every query. We also measured the time it took to retrieve the set of results for a query and then took a mean average value.

# Results

The following were the results that are plotted in the graph

We will now have a look at the run time performances of these systems



The run times also did not vary very much with the length of the queries.

VI. Conclusion

After performing evaluation of these systems, the overall results indicate that the BM25 performed significantly better than the ranked retrieval. The evaluation required careful examination of every test document especially while calculating recall since the recall also requires content of every relevant document that has not been retrieved. Using the same set of queries for evaluating these systems ensured fair measurement for Precision, Recall as well as for run time evaluation. The F-measure is calculated here by computing the Harmonic mean of Precision and Recall. The reason why it is Harmonic mean and not Arithmetic mean or Geometric mean is that Harmonic mean always has the least value compared to the other two. Thus, if we had a lower value of precision and a very high value of Recall, the Arithmetic mean would have been in an intermediate range whereas the Harmonic mean will always be closer to the smaller value. This results in a relatively more accurate measurement.

Thus we can assert that from these evaluation results, BM25 is a relatively better search engine than the Ranked model. However, the performance can be certainly improved in a variety of ways and is not just dependent on score computation this is also one of the reasons why Google is better than Yahoo or Bing and it is not just the score computation.

**References:**

1. http://en.wikipedia.org/wiki/Information\_retrieval
2. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, *Introduction to Information Retrieval*, Cambridge University Press. 2008.

M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.