CMSC5702 Parallel and Distributed Systems

Programming Project: Hadoop TF-IDF

Due May 1, 2015, 11:30 pm

1 Introduction

In this programming project, you will use Hadoop's implementation of map-reduce to search books. Given a keyword, you should retrieve the most relevant book. This is not a course in search, so the algorithm focuses on simplicity rather than search quality or performance. The most common algorithm to reflect how important a word in a collection or corpus is called TF-IDF, short for Term Frequency-Inverse Document Frequency. We will use TF-IDF to rank the search result.

2 Description

We introduce background knowledge on TF-IDF, your tasks in the assignment and how to make preparation before you get started.

2.1 TF-IDF

Term Frequency (TF). Suppose we have a set of English text documents and wish to determine which document is most relevant to the query "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document and sum them all together; the number of times a term occurs in a document is called its term frequency.

Inverse Document Frequency (IDF). However, because the term "the" is so common, this will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less common words "brown" and "cow". Hence an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely.

TF-IDF. TF-IDF is the product of two statistics, term frequency and inverse document frequency. TF-IDF is a well-know information retrieval algorithm for giving weights for terms

($t \in T$) of documents ($d \in D$). There are a number of variants of calculating TF-IDF. The algorithm we consider in the assignment is described as follows:

```
tf(t, d) = (number of t in d) / (total number of terms in d)
idf(t, D) = log((number of documents in D) /
                (number of documents containing t))
tfidf(t, d, D) = tf(t, d) * idf(t, D)
```

2.2 Tasks

Implement a map-reduce program `TFIDF.java` that finds the file name with the highest TF-IDF score. Your program should take several input parameters: the name of query file, a directory that contains the input books, and the output file name. The book files and query file can be downloaded from `http://www.cse.cuhk.edu.hk/~mssun/cmsc5702/asgn_data.tar.gz`.

Your program should be run using the following command:

```
$ hadoop jar tfidf.jar TFIDF query.txt books_repository output_filename
```

Note that the arguments to the command must all be located inside HDFS. The `tfidf.jar` file, on the other hand, is in the local filesystem.

We will provide `query.txt` file and books repository. `query.txt` contains search keywords separated by newline character(`\n`).

```
$ hdfs dfs -cat query.txt
abandon
ability
...
```

The books repository contains 25 English text books downloaded from Project Gutenberg.

```
$ hdfs dfs -ls books
Found 25 items
-rw-r--r--   3 cmsc5702 supergroup 167518 2015-03-04 17:01 books/pg11.txt
-rw-r--r--   3 cmsc5702 supergroup 305864 2015-03-04 17:01 books/pg232.txt
...
```

The output (i.e., query results) should consist of lines containing the search keyword, TF-IDF score, and followed by the file name of the book.

```
$ hdfs dfs -cat output.txt
abandon 0.00012256 pg5200.txt
ability 0.00012127 pg23.txt
...
```

Hints. Computing TF-IDF with MapReduce may contain multiple MapReduce jobs. For example:

- Job 1: compute term frequencies
- Job 2: compute number of documents each word occurs in
- Job 3: compute TF-IDF

2.3 Preparation

Because our project needs to run map-reduce on Hadoop, you need to first set up a Hadoop testbed inside a virtual machine (e.g., Ubuntu 14.04 in VirtualBox). We recommend you follow the Hadoop documents on setting up a single node cluster.

You may also find it helpful to start from the Hadoop MapReduce tutorial, which walks through a simple word-count example. After completing the tutorial, you can then adapt the tutorial code to complete the assignment.

3 Submission

Checklist:

- `TFIDF.java`: source code
- `output.txt`: query results
- `README.txt`: explanation of how your code works and why it is correct

Zip your files with the filename {`Student ID`}`.zip`. And send it to the tutor (email: `mssun+cmsc5702@cse.cuhk.edu.hk`) before the deadline. Note that late submissions will not be considered for this assignment.

4 References & Recommended Materials

- Wikipedia (TF-IDF): `http://en.wikipedia.org/wiki/Tf-idf`
- Hadoop Documents (Setting up a Single Node Cluster): `http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html`
- Hadoop Documents (MapReduce Tutorial): `http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html`