Big Data Search Engine

using Hadoop MapReduce

Amine Trabelsi

Submitted for Big Data
Assignment 2

a.trabelsi@innopolis.university
group DS-01
Innopolis University
2025

Contents

Methodology

Pipeline Overview

The pipeline consists of two main stages:

1. **Map Stage (mapper1.py)**: Tokenize each document and emit `(term, doc_id)` pairs.
2. **Reduce Stage (reducer1.py)**: Count the frequency of each term in each document, and store the result in the Cassandra table `inverted_index`.

Input Format

The input to the MapReduce job is a text file stored in HDFS (e.g., `/index/data`), where each line represents one document and follows the format:

```
<doc_id>\t<title>\t<text>
```

Mapper Logic

For each line in the input:

- Split the line into `doc_id`, `title`, and `text`.
- Tokenize the `text` into words using a regular expression (e.g., only alphanumeric terms).
- Emit a key-value pair for each word:

  ```
  <term>\t<doc_id>\t1
  ```

  This output represents a single occurrence of a term in a document.

Reducer Logic

The reducer receives all `(term, doc_id)` pairs and performs the following:

- Aggregates the frequency of each term within each document.
- Writes the result into the Cassandra table `inverted_index` with the schema:

  ```
  (term TEXT, doc_id TEXT, tf INT, PRIMARY KEY (term, doc_id))
  ```

Query Processing with BM25 Ranking

After the inverted index is stored in Cassandra, a PySpark application (`query.py`) is used to process user queries and retrieve the top 10 most relevant documents using the BM25 scoring algorithm.

Workflow:

1. The user query is provided via command-line argument or standard input.
2. The script connects to Cassandra to read:
   - Term frequency (`tf`) data from the `inverted_index` table.
   - Inverse document frequency (`idf`) values from the `vocabulary` table.
   - Document lengths and titles from the `documents` table.
3. Using the PySpark RDD API:
   - The BM25 score is computed for each document containing any of the query terms.

- Scores are aggregated per document.
- The top 10 documents are selected based on the total BM25 score.

4. The document IDs, titles, and BM25 scores are printed as output.

Execution: The application is run using a shell script `search.sh` that submits the PySpark job to the YARN cluster in fully distributed mode:

```
bash search.sh "example query"
```

Demonstration