report

Laith Nayal

April 2025

1 Methodology

This project implements a distributed search engine using Hadoop MapReduce for indexing, Spark RDD for ranking (BM25), and Cassandra as the persistent index store. The methodology is divided into distinct components:

1.1 Data Preparation

A Parquet file containing Wikipedia articles was processed using PySpark. The goal was to extract 1000 articles and save them in a unified format. Each article was saved as a plain `.txt` file using the format `<doc_id>_<title>.txt`, with spaces replaced by underscores. The contents were stored in HDFS under `/data`, and a clean tab-separated file was generated at `/index/data` with each line formatted as:

```
<doc_id>\t<doc_title>\t<doc_text>
```

1.2 Indexing using Hadoop MapReduce

Two MapReduce pipelines were used:

Pipeline 1: Inverted Index and Document Frequency (DF)

- **Mapper1**: Tokenizes document content and emits `(term, doc_id)`.
- **Reducer1**: Aggregates document IDs per term, calculates DF, and stores results into:
  - `inverted_index(term, doc_id)`
  - `term_stats(term, df)`

Pipeline 2: Document Length Statistics

- **Mapper2**: Emits `(doc_id, 1)` for each token in the document.
- **Reducer2**: Aggregates the total tokens per document and stores:

- `doc_stats(doc_id, doc_len)`

All data is stored in Cassandra tables under the keyspace `user12_keyspace`.

1.3 Ranking using Spark RDDs (BM25)

A PySpark application (`query.py`) reads the user query from standard input and computes BM25 scores for matching documents using the following formula:

$$idf(t) = \log\left(\frac{N - df_t + 0.5}{df_t + 0.5} + 1\right)$$

$$score(d, q) = \sum_{t \in q} idf(t) \cdot \frac{tf_{t,d} \cdot (k_1 + 1)}{tf_{t,d} + k_1 \cdot (1 - b + b \cdot \frac{len_d}{avgdl})}$$

Where:

- N: Total number of documents
- df_t: Document frequency of term t
- $tf_{t,d}$: Term frequency in document d (set to 1 in this simplified version)
- len_d: Document length from `doc_stats`
- $avgdl$: Average document length over all documents
- $k_1 = 1.5$, $b = 0.75$

1.4 Cassandra Schema

Data is persisted in Cassandra with the following schema in the `user12_keyspace`:

- `inverted_index(term TEXT, doc_id TEXT, PRIMARY KEY (term, doc_id))`
- `term_stats(term TEXT PRIMARY KEY, df INT)`
- `doc_stats(doc_id TEXT PRIMARY KEY, doc_len INT)`