UNIVERSIDAD EAFIT
INGENIERÍA DE SISTEMAS
ST 0263 TÓPICOS ESPECIALES EN TELEMÁTICA, 2020-2
GRUPOS 001 Y 002

PROYECTO2 – HPC

https://github.com/ctezna/horus-text-miner.git

Andres “el apa” Pulgarin Rodriguez apulgar3@eafit.edu.co
Carlos Tezna Sanz cteznas@eafit.edu.co
Nicolás “La salvación del pueblo” Gonzalez Vallejo ngonza27@eafit.edu.co

Introduction

Text mining is the use of various technologies in order to explore a big quantity of documents. This application helps the end user understand the content of a collection of texts through searching and indexing algorithms [3]. High Performance Computing (HPC) implies the use of all the computational and calculating power of the machine in order to achieve specific objectives.

Problem

Lack of efficiency in the text mining process. An example case of this problem is explained as followed:

> *A biology researcher has a large collection of papers/documents and they need to easily search and rank based on keywords or queries.*

Objectives

- Capable of receiving any collection of documents (pre-formatted to the input structure)
- Create a sequential program that creates an inverted index table based on the collection of documents (uses TF-IDF weighting) [1]
- Suggest documents based on the query provided by the user (Language Model OR Vector Space Model) [5][3]
- Develop a parallelized version of the sequential algorithm

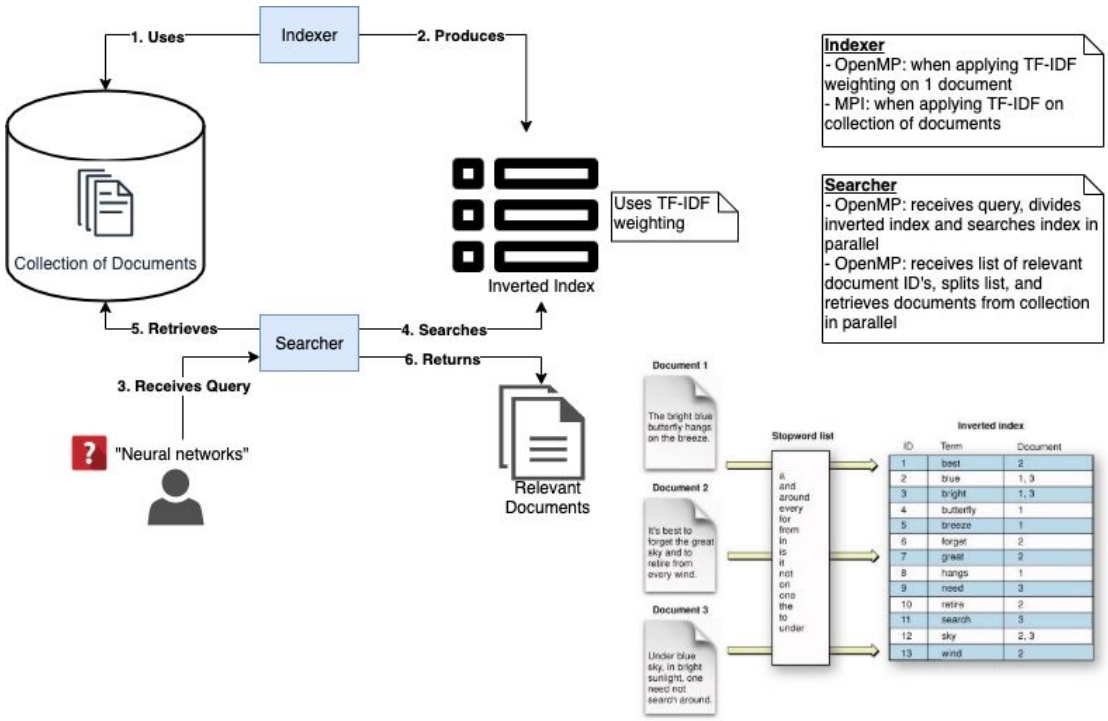

Figure 1: High-level view of Horus

Technical Requirements

- C++ or Python programming
- Information/Text retrieval theory
    - **Vocabulary:** All the terms in the document collection.
    - **Bag of Words:** A strategy of resembling documents where a document is represented as a collection of terms and their frequencies. Order of the terms is disregarded.
    - **TF-IDF**: Term frequency - inverse document frequency, is a metric used to measure the occurrences of terms while also measuring the frequency of the term across the entire collection. This provides a higher ranking for terms that are more significant to a certain document. (The word *"the"* frequently appears in all documents, making it less relevant to one specific document.)
    - **N-gram:** A sequence of *n* terms used to represent phrases that are naturally occurring (ex. "White House", "computer science", "exploratory data analysis"). Often in practice, bigrams or trigrams are used. (Helps combat the bag of words approach of no word ordering)
    - **Inverted Index**: Table-like key value data structure that has a term as the key and a list of document IDs sorted based on TF-IDF score as the value.
    - **Vector Space Model:** A similarity function that relates the user query with documents based on a distance measure. [4]
    - **Language Model:** A probabilistic ranking function based on Bayes Rule to order documents decreasingly by their estimated probability of relevance. A popular implementation of this algorithm is BM25. [4]
    - **Smoothing:** Used in combination with a probabilistic ranking function so that no probability is zero. Also known as adding a *dummy count* to each term. Laplace smoothing, Jelinek-Mercer smoothing, or Dirichlet smoothing are considered here (see Tradeoffs section). [4]
    - **Stemming/Lemmatization:** Reducing words to their base or root word. ({"computing", "computed", "computes", "computer"} => "compute")
- Parallelization using OpenMP and MPI
- Data management

Tradeoffs

- **Sequential Execution vs. Parallel Execution**

  These two forms of program execution provide an obvious tradeoff where a sequential execution would be easier to implement, and where a parallel execution would provide higher efficiency.
- **Use of N-grams vs. No use of N-grams**

  The use of N-grams allows for representation of phrases, especially those where the individual terms have little to no relation with each other (ex. "Burger King").

N-grams require a combination of every term in the vocabulary (grows the vocabulary exponentially). If no n-grams are used, the program will be computationally less expensive but will assume all terms are independent of each other. [2]

- **Vector Space Model vs. Language Model**

  The vector space model (VSM) is a vector representation that uses a geometric/linear algebra based approach. VSM is a simpler approach and easier to interpret. Language models are probabilistic approaches that often provide more accurate rankings, but are more complex in implementation.

- **Laplace smoothing vs. Jelinek-Mercer smoothing vs. Dirichlet smoothing**

  "... add one to all the bigram counts, before we normalize them into probabilities... This algorithm is called Laplace smoothing. Laplace smoothing does not perform well enough to be used Laplace smoothing in modern n-gram models, but it usefully introduces many of the concepts that we see in other smoothing algorithms, gives a useful baseline, and is also a practical smoothing algorithm for other tasks like text classification." [2] Dirichlet performs better for keyword queries, Jelinek-Mercer performs better for verbose queries. [4]

Work Plan

1. **Create document collection:** Extract and clean documents from multiple sources.
2. **Build Indexer component:** Develop a program that uses the collection of documents and produces an inverted index table.
3. **Build Searcher component:** Develop a ranking function that receives the user input and using a searcher, returns relevant documents to the user.
4. **Baseline testing:** Execute the program using the sequential implementation.
5. **Parallelization:** Develop a parallelized approach.

Conclusions

Although text mining is a well known process to gather information of numerous documents, it can often be a slow task to perform. With the help of technologies derived from High Performance Computing this same process can be successfully achieved in a shorter time. The **Horus Text Mining framework** aims at providing a high performant and lightweight program for text retrieval and mining.

References

[1] Grossman D.A., Frieder O. (2004) Parallel Information Retrieval. In: Information Retrieval. The Kluwer International Series on Information Retrieval, vol 15. Springer, Dordrecht. https://doi.org/10.1007/978-1-4020-3005-5_7

[2] Jurafsky, D., & Martin, J. H. (2019). *N-gram Language Models* [PDF]. Stanford: University of Stanford.

[3] Loiacono, D. (2010). *Information Retrieval & Text Mining* [PPT]. Milan: Politecnico di Milano.

[4] Pecina, P. (2017). *Probabilistic Models for Information Retrieval* [PDF]. Maceió, Brazil: Universidad Federal de Alagoas.

[5] Waterloo, U. (2010). *Parallel Information Retrieval* [PDF]. Boston: MIT Press.

Código de Honor:

Cada uno de los autores, debe declarar explícitamente que el trabajo es original y cuál fue su aporte en el desarrollo del proyecto 2, o si es copiado de algún sitio en internet (porque hay muchas implementaciones de este problema) deberá referenciar o citar el sitio de donde tomó el trabajo y declarar entonces cuál fue su aporte con esta copia en el proyecto 2.

Esta declaración debe ser colocada en README.md del github del proyecto 2 por cada autor.