# Assignment 7 Report

Rohan Chaturvedi, Anany Mishra, Saarth Singla, Pratyaksh Dhairya Panwar

November 15, 2023

## 1 Introduction

As part of Assignment 7, we have developed a code representing a text analysis and question-answering system. This report aims to elucidate the various data structures and algorithms incorporated within the code.

## 2 Code Components

Our code consists of several key data structures and algorithms as mentioned below:

### 2.1 Trie and AVLMap Implementation

Our code features custom implementations of Trie and AVLMap, which are essential data structures employed for efficient word storage, counting, and retrieval within the text analysis system.

The Trie data structure is used by us to store and manage words encountered in the text. Each node of the Trie represents a character in a word. This structure enables efficient word retrieval and allows for the storage of word counts per paragraph using the AVLMap.

Within each Trie node, there exists an AVLMap. This AVLMap stores the count of a particular word in each paragraph. For instance, when encountering a new word in a paragraph, the code navigates the Trie to find or create the corresponding node and updates the count of that word within the AVLMap associated with that node.

Additionally, the Trie nodes maintain a total count of occurrences for each word encountered in the text. Moreover, the initial count values for words are derived from the CSV file provided.

### 2.2 RAKE Algorithm

The RAKE (Rapid Automatic Keyword Extraction) algorithm within the code plays a pivotal role in extracting keywords or significant phrases from textual data. Here is a detailed breakdown of its implementation:

The RAKE algorithm initiates by segregating stop words from the text. A predefined list of stop words, comprising common language elements such as articles, conjunctions, and prepositions, is filtered out from the text. This step is crucial to focus on meaningful content and disregard commonly occurring, less informative words.

Once the stop words are eliminated, the algorithm iterates through the remaining text, isolating words and accumulating them into a list. This list undergoes a sorting process to organize the words alphabetically, facilitating subsequent processing steps.

Following the sorting process, the algorithm conducts frequency analysis to determine the occurrence of each word.Subsequently, the algorithm identifies keywords or significant phrases by discerning consecutive unique words or phrases with varying occurrences. When a change in word or phrase occurrence is detected, the algorithm recognizes the boundary of a potential keyword or phrase. These segments, often associated with higher occurrence counts, are earmarked as keywords or key phrases within the text.

## 2.3 TextRank Algorithm

Our text analysis and question-answering system employ the TextRank algorithm to rank pages based on their significance and relevance. Here is a detailed overview of its implementation:

The TextRank algorithm operates by interpreting each paragraph within the textual data as a node within a graph. The connections or edges between these nodes represent the relationships between paragraphs. This graph-based representation enables the algorithm to compute the importance of each page by considering its interconnectedness with other pages.

Initially, the system employs the TF-IDF (Term Frequency-Inverse Document Frequency) algorithm to identify and select the most significant paragraphs from the textual data. TF-IDF calculates the importance of words in each paragraph relative to a corpus of documents, enabling the system to pinpoint paragraphs with distinct and meaningful content.

Following the TF-IDF selection, the TextRank algorithm processes the significant paragraphs. It considers the interconnectedness between these paragraphs and iteratively computes the importance of each page. Pages that have numerous connections from other important pages are assigned higher scores, signifying their relevance and centrality within the textual structure.

The final scores obtained from the TextRank algorithm effectively rank pages based on their importance and relevance within the context of the given textual data. These ranked pages serve as a valuable resource during the question-answering process, aiding in the retrieval of pertinent information based on their calculated significance.

## 2.4 Question-Answering Logic

Our code handles user queries by analyzing text, identifying relevant paragraphs, and presenting them as potential answers. The best 'k' value is chosen based on the variance and mean of the given ranking scores.

## 2.5 Heap Data Structure

We have used Heaps for maintaining top-k results and for sorting operations to efficiently retrieve relevant paragraphs.

# 3 Conclusion

In summary, our code represents a comprehensive approach to text analysis and question-answering. It employs various algorithms and data structures to process text data efficiently, extract relevant information, and provide answers to user queries.