COSC 320 – 001

Analysis of Algorithms

2022/2023 Winter Term 2

Project Topic Number: 1

Keyword Replacement for Text

Group Lead: Brandon Mack

Group Members: Aidan Elliott, Eddy Zhang

For Project Proposal use the following headers. (Maximum 2 pages excluding the cover page)

Problem Description.

In natural languages, abbreviations are used to shorten writing while maintaining meaning within the statement. In the digital age, new slang terms and abbreviations are fabricated or fazed out quicker than ever thanks to social media. You would be forgiven for not understanding a tweet that read "TFTF! ICYMI, I sent you a CC DM of the TT you mentioned TOD!" Or maybe you are reading about a rare medical disease with an abbreviated name. Our program strives to remove this confusing struggle by translating these abbreviations for you. We propose that by inputting each word into a large Hashmap containing the abbreviations as the key and and its corresponding phrase as the value, we can efficiently translate the tweet for the reader in real time.

Edge Cases.

1. **List of abbreviations is empty**
    - If the number of elements in the abbreviation list is zero, then no comparisons are needed. We could use an if statement to check for the abbreviation list's size and terminate the program if size = 0. Time complexity will be O(1).
2. **Tweet that has 0 words**
    - Similarly, if a tweet has no words in it, no comparisons are needed. Time complexity will be O(1).
3. **List of abbreviations of size 1**
    - If the number of elements in the abbreviation list is 1, then the program will achieve O(1) lookup from the abbreviation list even if we use an array instead of a Hashmap. The overall time complexity will be O(m), where m is the number of words in a tweet.
4. **Tweet that has 1 word**
    - Time complexity will be O(1) if a Hashmap is used or O(size of the abbreviation list) if an array is used to store the list of abbreviations.

Expected complexities.

The Brute force method would involve creating a list of all possible abbreviations and linearly searching it. If we have n possible abbreviations and a tweet with m words, translating the tweet would take O(nm). For our improved method, we know that you can search a Hashmap in O(1). Since we are searching the Hashmap for every word, translating a tweet with m words would take O(m).

Dataset Collection. Include plans for dataset collection or generation. If you already have searched for it, include links to the dataset. Also, mention the details of the dataset: how many records, what columns (fields), format of the data, etc.

Our plan is to find a dataset which already contains hundreds of abbreviations and their corresponding phrases. With some light googling, no clear option has become immediately available. If this continues, we could manually create our own dataset by either manually inputting into a csv file or automate it using some form of web scraping.

Programming Language. Java

Task Separation and Responsibilities. We plan to divide up the responsibilities equally and work together closely rather than branching off and doing our own thing. The main focus will be meeting together and working through the issues together.

Unexpected Cases/Difficulties.

- If someone has used a keyword that is not within the data set we will be unable to identify and perform a phrase replacement. A solution to this would be to have a feature that adds the abbreviation to the dataset similar to adding a word to a dictionary in Microsoft Word. Implementing this may require our algorithm to do some form of web scraping, which is probably outside the scope of this project. Our best solution would be to create a large enough dictionary that covers all forms of topics from slang terms to medical and financial abbreviations.