UNIVERSITY OF MARYLAND BALTIMORE COUNTY

DEPARTMENT OF COMPUTER SCIENCE

ADVANCED OPERATING SYSTEMS

CMSC 676

PROJECT 1

SUBMITTED BY:

SUSHMITHA MANJUNATHA

INTRODUCTION

In this project, I have tried to implement a n node tokenizer that takes input HTML files and it will parse it to token and sorting tokens based on tokes as well as sorting it based on frequency

SYSTEM DESIGN

- I have implemented a program to tokenize the documents. Additionally, I have compared the performance of my approach with that of another approach in terms of quality of tokens and running time.
- I have used Python to code the program. To execute the program from a Linux terminal

INPUT

- A directory of input documents is given to the program as input from the command line.
- The documents were html files with primarily ascii content and with some unicode as well.
- The files were named using three digit numbers starting from 001.

OUTPUT

- The output from program is written to the directory called output_files.
- This directory contains two sub directories in it named 'token_files' and sorted_files.
- The first one contains all the tokenized documents with one output file per input file.
- Each of these files are named as 'input_file_name'.txt, for example the output token file for 001.html is 001.txt
- The content of these files are follows, Token : frequency_of_occurence
- The second directory contains two files with the complete vocabulary sorted by token and frequency.

The Document

```
<html>
<body>
<h1>Title</h1>
<p>A <em>word</em></p>
</body>
</html>
```

The DOM Tree

```
DOCUMENT
└─ELEMENT: html
   ├─TEXT: '\n'
   ├─ELEMENT: body
   │  ├─TEXT: '\n'
   │  ├─ELEMENT: h1
   │  │  └─TEXT: 'Title'
   │  ├─TEXT: '\n'
   │  ├─ELEMENT: p
   │  │  ├─TEXT: 'A'
   │  │  └─ELEMENT: em
   │  │     └─TEXT: word
   │  └─TEXT: '\n'
   └─TEXT: '\n'
```

METHODOLOGY:

I used Python to program the tokenizer. I made use of python libraries such as Counter from collections, BeautifulSoup, re, Codecs and Glob to clean up the text and to build the vocabulary. Accepting the input and output file names from command line.Read and Parse the text data and removing HTML encoding by Beautifulsoup package.

BeautifulSoup: BeautifulSoup is a Python package that parses *broken* HTML, HTML parsing is easy in Python, especially with help of the BeautifulSoup library.when compared to other HTML parsing libraries this is more efficient.

Counter from Collections: The counter class itself is a dictionary subclass with no restrictions on its keys and values. The values are intended to be numbers representing counts, but you *could* store anything in the value field.

To count the occurrences of each word in the list I used the Counter function from collections library in Python. It takes a list as input and returns a dictionary (Python object) with tokens as keys and their frequency as values.

Counter is used in the program to count the frequency of each token

Codecs: standard Python codecs (encoders and decoders) and provides access to the internal Python codec registry which manages the codec and error handling lookup process. Codec is to specify encoding of the html files

GLOB: The **glob** module finds all the pathnames matching a specified pattern according to the rules used by the Unix shell, although results are returned in

arbitrary order. No tilde expansion is done, but *, ?, and character ranges expressed with [] will be correctly matched.

Glob is used to find the path name of input files directory

MatplotLib: Used For Plotting the graph and performace analysis

Time: Time library is used to measure the time and the function time.time

Functions used :

Get_text(): This function is imported from beautiful soup,used to obtain text from the file.

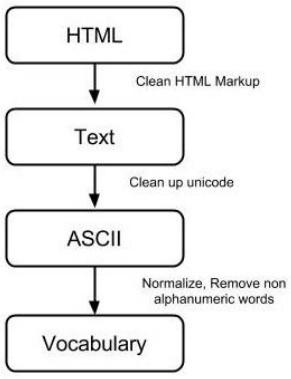
Sorted(): Sorted function of python is used to sort the token on the basis of tokens and frequency of occurance of tokens.

Fillter(): Filter function along with regular expressions in used in the program to chopout the unwanted spaces,tab,special charaecters,numbers.

Findall(): Findall function from regular expression is used to find all substrings where the Regular Expression matches and returns them as list**.**

Lower(): this function is used to downcase all the words to lower case letters.

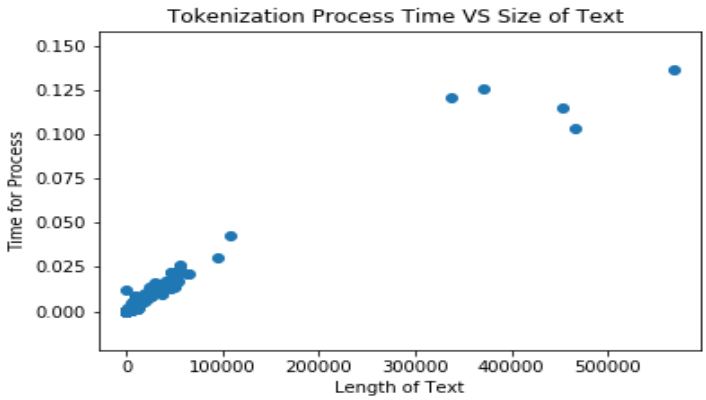
The following diagram shows the control flow with respect to various modules



STEPS of Tokenizer

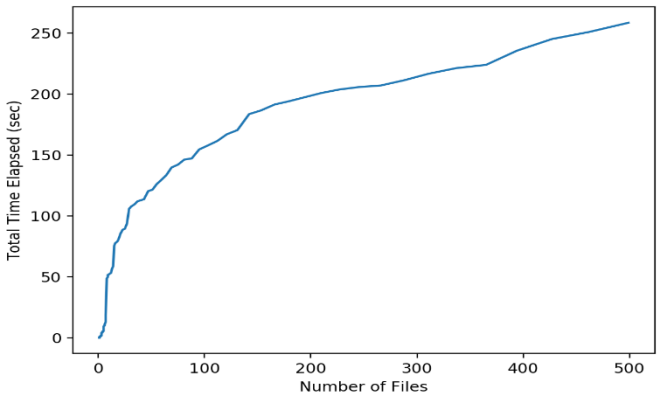
1. Read the name of the directory from command line
2. Open the directory
3. Read the files
4. Load the data
5. Sort the data by length
6. Convert the words into lower case
7. Handling puncuationst,numbers,special characters-Used Regular Exp
8. Count the frequency of occurrence of each of the tokens
9. Sort by value and Sort by frequency-Counter is used
10. Save to disk

EVALUATION: The following graph shows the running time of the program Tokenizer X .As you can see from the graph both the system time and elapsed time increases almost linearly. I compared the results with the program of Tokenizer Y. The efficiency of his program has been shown below in the graph. The output of both tokenizers were similar in quality as both of us had taken care to clean the meta data as well as converting unicode characters.



COMPARISION RESULTS: After comparing the results between Tokenizer A and Tokenizer B, we can see that the time taken by both the tokenizers grow linearly with respect to the number of files. Tokenizer A takes more time than Tokenizer B for the same number of files. One of these might be a probable cause for this difference in method used for tokenizing.

Graph of tokenizer A: which HTMLparser library to Parse HTML files



CONCLUSION: As BeautifulSoup is a library lighter than NLTK, it consumes less memory and executes faster. BeautifulSoup as it is easy to use. Additionally it provides a suite of tokenizing functions for handling various types of text.

NOTE:There are two python files one is for linux version of code and another is for windows