# Project Description

This file contains information on how the special cases are handled while generating tokens out of Cranfield document collection and the output is included.

Algorithm:

1. All the files in a given folder are read by using listFilesForFolder() function.

2. Of the total list, one file is processed at a time using the below steps:

1. The lines starting with the SGML files are ignored.
2. A line of text is read and is split using the space as the delimiter.
3. The resulted token words are passed to token processor function for further processed using *processWord*() function.
4. When *processWord*() returns the finalized tokens, they are stored in the *tokenCollection* (of TreeMap type) along with their frequency in the files.
5. Now, call stem function on each token and store the returned word stems into *stemCollection*. - If the word stem is encountered for the first time, insert it into the *stemCollection* along with frequency

4. Now, sort these two maps based on the value field, to retrieve top 30 frequent words or distinct words.

***processWord*() function:**

The initial word identified using space delimiter is sent to this function for further processing. The following actions are performed on the received word.

1. If the word is of abbreviated form (ex: U.S.A.), then the dots are kept as is. (U.S.A.).

2. If the word is of possession form (ex: University’s), the postfix ‘s is removed (University).

3. If the word contains hyphens in the middle (ex: multi-layer), hyphens are removed to form a single word (multilayer).

4. If the word contains ‘- -‘ in the middle, then ‘- -‘ are removed.

6. If the word contains ‘,’ in the middle(“schidler,j”), then it is removed and replaced by space which is in turn treated as 2 tokens(“schindler”, “j”).

7. All the digits are removed from the words. Therefore the alpha numeric words like a2b3c4 becomes abc.

8. If the word contains any other special characters after processing these will be replaced by empty char.

Program-Overview:

1. Running Time: The program took 1638 ms to acquire the text characteristics during the test run.

2. Handling special cases while tokenization:

a. Upper and lower case words: All the tokens are converted into lower case words, so that the words “Apple”, “aPple” and “apple” do go into the same entry(“apple”).

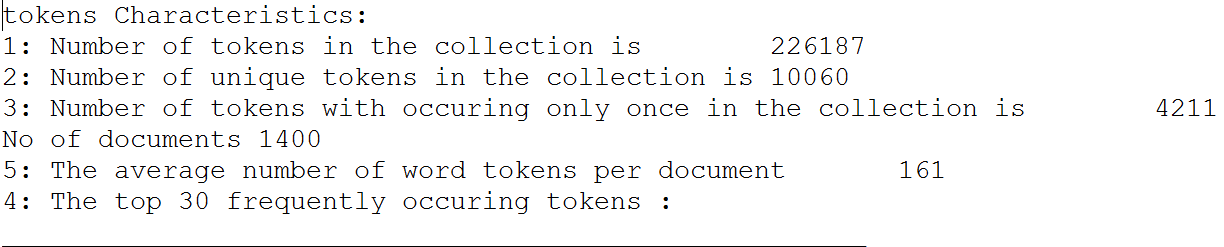
b. Words with dashes: words having the dash in the middle (“30-year”) are transformed by removing dash(“ 30year”)

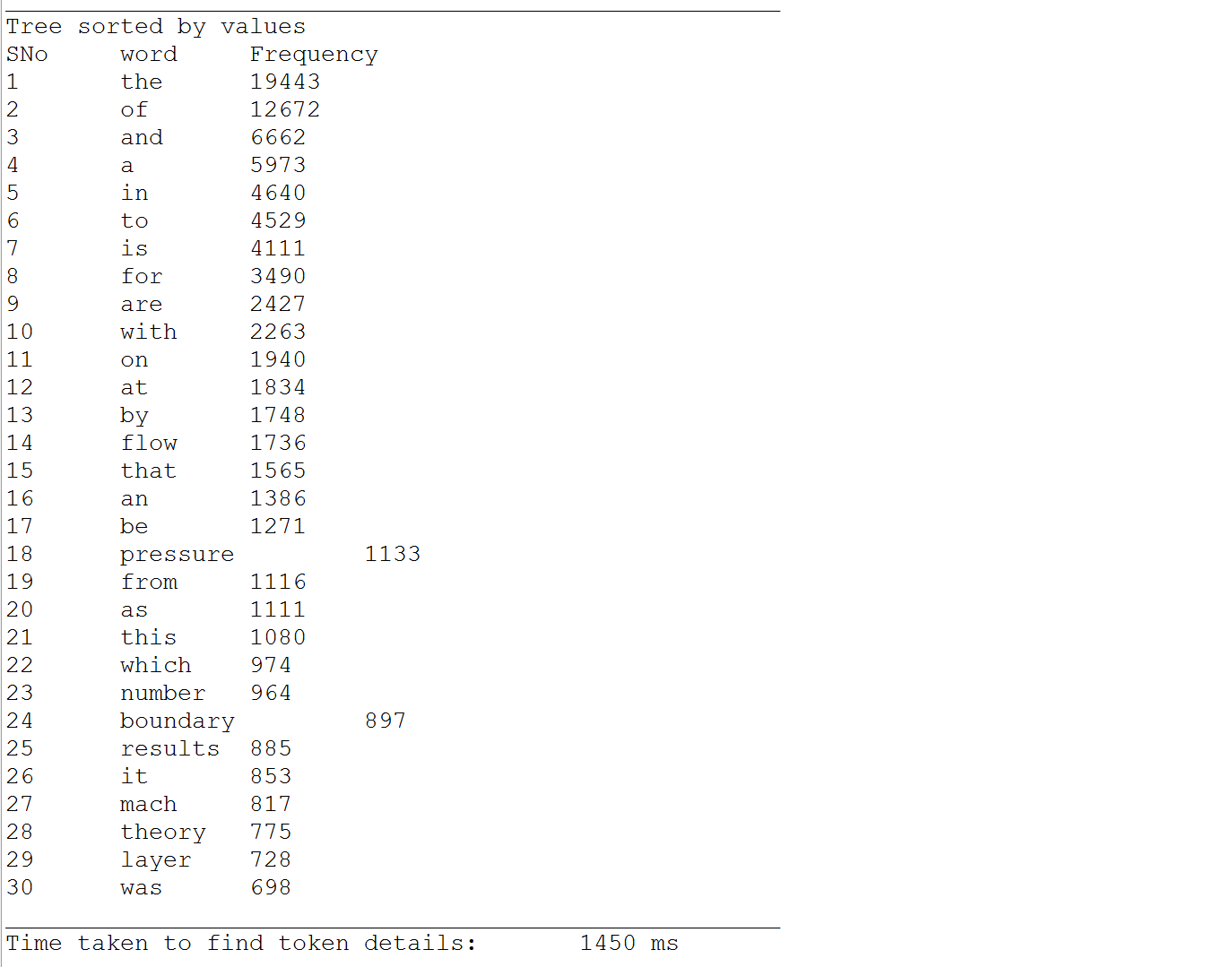
c. Possessives: Possessive words (“schindler’s”) are transformed to non-possessive words by truncating ‘s at the end(“schindler”).

d. Acronyms: Acronyms (“U.S.A”) are transformed by removing dots (“u.s.a.”).

3. Major Algorithms and data structures: algorithms is described in the program- description above. TreeMap is used to store tokens and stems as insertion, retrieval and comparison takes linear time.

Final Statistics of Tokens :





Final Statistics of Stems:

