

# Lab3 Part 2 TNM098 – Text Comparison

Jens Jakobsson and Jonathan Bosson

## Introduction

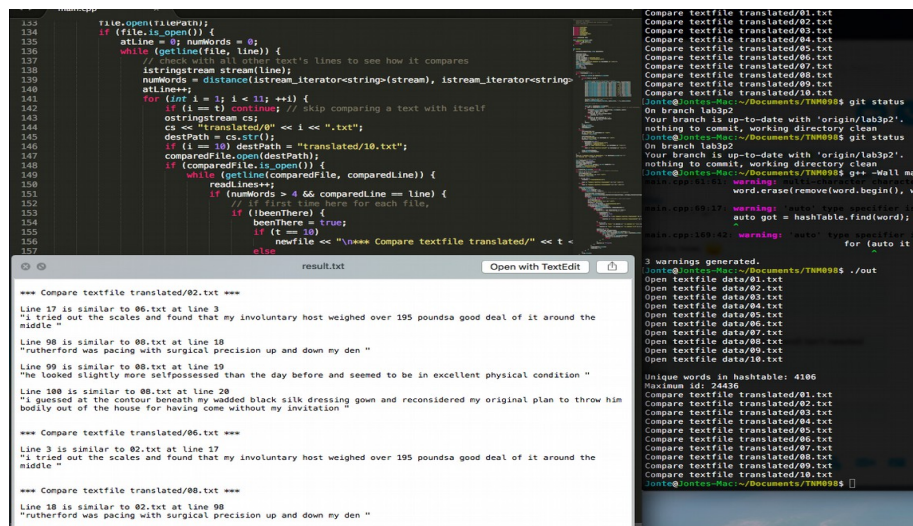
Given 10 different texts of varying length the task was to write a program to detect the files that contain plagiarised content.

## Method

It's a smart idea to convert a word into a unique integer number to let the computer more efficiently compare texts. Therefore, the first thing the program does is read in each textfile and builds a dictionary from the words used in the different texts. Furthermore does it create additional translated texts that contain a bunch of numbers with spaces and new lines in between. Each number sequence is an ID for a specific word in the dictionary and each line stands for each sentence.

Once finished, the program will open each textfile seperately and read line by line. For each line that is read, it will scan through all lines in the other translated texts to find similarity. As of this solution, the complete sentence has to be identical for the program to detect the plagiarism. A check for minimum amount of words can be put in by the user so short sentences don't clutter the result. Once a match has been found, the program will write in what files and what lines are similar into a new textfile as well as the sentence which a match was found for.

## Result



```
132 file.open(iterator);
133 if (file.is_open()) {
134     atLine = 0; numWords = 0;
135     while (getline(file, line)) {
136         // Check with all other text's lines to see how it compares
137         istream_iterator<string>(stream), istream_iterator<string>(
138             numWords = distance(istream_iterator<string>(stream), istream_iterator<string>(
139             atLine++;
140         for (int i = 1; i < 11; ++i) {
141             if (i == 1) continue; // skip comparing a text with itself
142             ostreamstream cs;
143             cs << "translated/" << i << ".txt";
144             destPath = cs.str();
145             if (i == 10) destPath = "translated/10.txt";
146             comparedFile.open(destPath);
147             if (comparedFile.is_open()) {
148                 while (getline(comparedFile, comparedLine)) {
149                     readLines++;
150                     if (numWords > 4 && comparedLine == line) {
151                         // If first time here for each file.
152                         if (libenThere) {
153                             beenThere = true;
154                             if (t == 10)
155                                 newfile << "n*** Compare textfile translated/" << t <
156                             else
157                                 result.txt
158                                     Open with TextEdit
159                                     *** Compare textfile translated/02.txt ***
160                                     Line 17 is similar to 06.txt at line 3
161                                     "I tried out the scales and found that my involuntary host weighed over 195 pounds a good deal of it around the
162                                     middle "
163                                     Line 08 is similar to 08.txt at line 18
164                                     "rutherford was pacing with surgical precision up and down my den "
165                                     Line 09 is similar to 08.txt at line 19
166                                     "he looked slightly more selfpossessed than the day before and seemed to be in excellent physical condition "
167                                     Line 100 is similar to 08.txt at line 28
168                                     "I guessed at the contour beneath my wadded black silk dressing gown and reconsidered my original plan to throw him
169                                     bodily out of the house for having come without my invitation "
170                                     *** Compare textfile translated/06.txt ***
171                                     Line 3 is similar to 02.txt at line 17
172                                     "I tried out the scales and found that my involuntary host weighed over 195 pounds a good deal of it around the
173                                     middle "
174                                     *** Compare textfile translated/08.txt ***
175                                     Line 18 is similar to 02.txt at line 08
176                                     "rutherford was pacing with surgical precision up and down my den "
177                                     Unique words in hashtable: 4106
178                                     Maximum id: 24216
179                                     Compare textfile translated/01.txt
180                                     Compare textfile translated/02.txt
181                                     Compare textfile translated/03.txt
182                                     Compare textfile translated/04.txt
183                                     Compare textfile translated/05.txt
184                                     Compare textfile translated/06.txt
185                                     Compare textfile translated/07.txt
186                                     Compare textfile translated/08.txt
187                                     Compare textfile translated/09.txt
188                                     Compare textfile translated/10.txt
189                                     jont@jontas-Mac ~/Documents/TNM098$
```

The program takes roughly 9 seconds to run entirely with both building the dictionary and finding the plagiarised content on a low-end computer, Macbook Air 2011 model. The result varies a bit depending on what the user decides the minimum amount of words should be. If a sentence doesn't have to be longer than three words would a lot of similarities between most files be found due to common expressions or sentences. *"I/He/She shook her head"* was a particularly popular sentence throughout most files. The following table was the result with a minimum of five words.

*** Compare textfile translated/02.txt ***
Line 17 is similar to 06.txt at line 3 "i tried out the scales and found that my involuntary host weighed over 195 poundsa good deal of it around the middle"
Line 98 is similar to 08.txt at line 18 "rutherford was pacing with surgical precision up and down my den"
Line 99 is similar to 08.txt at line 19 "he looked slightly more selfpossessed than the day before and seemed to be in excellent physical condition"
Line 100 is similar to 08.txt at line 20 "i guessed at the contour beneath my wadded black silk dressing gown and reconsidered my original plan to throw him bodily out of the house for having come without my invitation"

## Discussion

This programs result is very binary in its search. It will either find an identical sentence in another file or say there are no similarities at all. A more thorough search could be implemented that reads word by word and through weights estimates how plagiarised a file is by percent. However, reading full sentences was a full valid solution for the set task.

Due to this, the program didn't need to read word by word and could compute more efficiently. Another efficient trick was using a hash table over a linked list for the dictionary. In terms of building the dictionary it is just as expensive to insert elements. However, finding a key is done in  $O(1)$  rather than  $O(n)$ . In this lab 24436 words were read but only 4106 unique ones were inserted into the dictionary, which is a common occurrence. This means the majority of the read words were not implemented since they were repetitions of previous ones, which made a data structure that can find elements quickly efficient.

A cost of this is when we need to decode a message. Since the hashtable uses *pair*<Key, item> there is no efficient way to find the key by searching by item. A  $O(n)$  search by stepping through the dictionary to find by item can still be done. Since there is only need to decode the ID back to the word if a match has been found, was this process not be done many times. In the table result that is 50 times out of 24436 read words. The product is a program that can render results quite quickly.