Intro

The goal of this project is to build a plagiarism detector by using algorithms studied in class to find the Longest Common Subsequence (LCS) between two input texts. The Longest Common Subsequence is the longest subsequence that occurs in both texts. A subsequence is defined as a sequence that can be obtained from another only by deletion of elements, and not reordering. The elements need not be consecutive in the original text.

These are the algorithms implemented to find the LCS between two texts (described in detail in section 2.1):

* Dynamic Programming
* Divide and Conquer (linear space)
* Recursive
* Recursive with Branch and Bound

The project also includes the implementation of algorithms studied in class to solve the printing neatly problem. The printing neatly problem refers to the problem of printing a text while minimizing the number of empty spaces at the end of each line. Formally, the text is composed of *n* words where the number of characters in each word are represented as *l1, l2, … , ln*. The text is to be printed so each line holds a maximum of *M* characters. The number of extra spaces printed at the end of a line which contains words *i* through *j* is defined as follows:

*M - j + i -*

We want to minimize the sum of the cube of this non-negative value over all lines but the last. This is implemented using the following algorithms (described in detail in section 2.2):

* Dynamic Programming
* Greedy
* Recursive with Branch and Bound

Lastly, a study of the behavior of LCS algorithms and printing neatly algorithms is presented, including runtime analysis, space analysis, experimental runtime analysis and comparisons, and scalability.