Project2: MDNA

Andrew Yang 11229916

# Project Description

This program is used to check for copyright infringement between two songs by determined the music notes. This program will check the alignment between two songs using a modified Needleman-Wunch algorithm. It will calculate and show the number of percentage difference between two songs, in order to determine the copyrights infringement happened or not. If the percentage is above the allowable similarity threshold copyright infringement has occurred.

# Data Structures

The structure “ConfigInfo” describing the configuration of the analysis, which include “(string variable “firstSongFile”)The first line will specify the name of the first song file”, “(string variable “secondSongFile”) The second line will specify the name of the second song file”, “(int variable “matchScore”) The third line will specify the score for a match”, “(int variable “mismatchScore”) The fourth line will specify the score for a mismatch”, “(int variable “gapScore”) The fifth line will specify the score for a gap”, “(int variable threshold) The sixth line will specify the allowed similarity threshold for copyright infringement”.

The structure “SongInfo” is the information of each song, which include name, idnumber, and originalMusiccalNote (all in string).

Void function “loadConfigurationData” use to load configuration data from file. Void function “loadSongData” use to load song data from file. Int function “analyzePercentageSimilarity” use to set the percentage of the similarity. Int function “calculatePercentages” use to calculate the percentage difference of two songs. Void function “split” use to split a string line by flag. String function “parseConfigurationRow” use to parse the configuration of the row.

In the main function, the program will load songs file and put all the information into my structures. After that, it will use all the functions which I have mention above to determine the percentage difference between two songs and show the result.

# System Functionality

Void function “loadConfigurationData” use to load configuration data from file. First, I use ifstream to open the file. Second, using the getline to get the configuration data from input file. In order to make confirmation of music note, I use getline to find the matchsocre, mismatchscore, gapscore, and threshold. At last, close the file.

Void function “loadSongData” use to load song data from file. First, I use ifstream to open the input file. Second, use getline to get the song information. Using vector to deal with arrays and update the information. At last, close the file.

Int function “analyzePercentageSimilarity” use to set the percentage of the similarity. First, I use vector to generate the matrix. The system must fill in the alignment matrix. All slots along the top row should be set to the gap penalty times their index, i.e. 0\*gap, 1\*gap, 2\*gap… (using for loop to get gapScore for 2 songs). Starting at the top left emptiest slot and working left to right, top to bottom (using for loop to get the col and row of the music note). If the music notes for each song corresponding to the slot’s position are the same, then the slot’s value should be the diagonal up and left slot’s value plus the match score (using if/else method to distinguish the information). Output the complete alignment matrix with appropriate top and left headers (using for loop to do the print). Next the system should trace back through the matrix to align the songs correctly. Start at the bottom right slot in the matrix and work towards the top left slot (using while loop to read): If the music notes corresponding to the current slot are the same then move diagonally up and to the left in the matrix and add both notes to their respective alignments (if else method). If the music notes corresponding to the current slot are not the same then you must determine whether a gap or mismatch has occurred. Your system should check diagonal, up, and left (in the order) and select the first direction who’s requirements have been fulfilled (if else method). If you ever end up along the left or top side of the matrix, add “-“ to the alignment of the song for which no more notes exist, and add the notes of the other song to the other alignment (using two while loop to do this action). Output the ID, name, and alignment of each song. Finally, return back to calulatePercentagesSimilarity.

Int function “calculatePercentages” use to calculate the percentage difference of two songs. First, I create an int variable “diff”. Second, Calculate the humming distance (using for loop to read the size). Calculation: (double rate)Percentage similarity rate= diff/size. I round the percentage similarity to the nearest whole percent (int result = (int)round(rate \* 100)). Finally, return back to result.

Void function “split” use to split a string line by flag. String function “parseConfigurationRow” use to parse the configuration of the row. These two functions using vector and update/putback the information.

In the main function, the program will load songs file and put all the information into my structures. After that, it will use all the functions which I have mention above to determine the percentage difference between two songs and show the result.