# **CSCE 5150 Analysis of computer Algorithms**

# **Project Update Report for LOSS LESS DATA COMPRESSION**

My name is Pragnya Gannapureddy, and I am doing this project together with Nitheesh Edla. Using the greedy approach method, we are implementing a lossless data compression algorithm so that we could encode and decode the string using binary codes. We designed our algorithm which looks similar to Huffman coding, we also almost completed the project and tested the different improvements that can be made.

No changes have been made in our motivation for the project and everything as of now is promising that we going to do it before the deadline.

**Datasets:**

We couldn’t find any datasets for this project online all we can find is only strings being used for testing the project, so we have created the text files for testing our project. We have generated files of different file sizes (files of 1KB, 1MB, 2MB, 4MB,……,256MB) for checking the time and space complexity.

**Algorithm used:**

We are using the Greedy approach to design an algorithm that does 4 tasks and by those 4 tasks, we are going to achieve our goal of lossless data compression.

The 4 tasks are:

1. Get frequencies of all the different characters that are present in the file and store them.

2. Build a tree and store all the characters based on the frequency generated earlier and assign binary code for each of the leaf and parent nodes.

3. Encode all the characters in the tree by putting all the binary values along the way together. (Eg: If we have a character at the 3rd level and on the right then the binary value of that character would be 01(if the parent of the character is left node) or 11(if the parent is also a right node).

4. Decode all the binary values into characters and rearrange the string.

**Testing the program:**

We have started testing the project once after completing the code and we got some hopeful results.

**Original string was :**

My name is Edla Nitheesh, and I am doing this project together with Pragnya Gannapureddy.

**Encoded string is :**

100101100111101010100111100010111110100111010100010111110001111110001000100111010000110001110101001000101111011011001111110000110101110010100101100111011111100100100110100111101010110001111100111010110010110001111100101101100000111010001010110101111001011101001111001101000100110010110011010111011110001010101010111111010110000101011111000101111100101110111101011111011000001100110000001100010101011011110011001111010001100011100101111101101010000111101

**Compilation time: 0.83 seconds**

**Time complexity of the Algorithm is O(nlogn).**

**Note: Couldn’t add other test cases because of the size of the report.**

**Experiments:**

We have conducted different types of experiments to check the correctness of the program like for example we have given alphabets in capitals, we have given different symbols and spaces, In the case of capitals, it is being considered as the same alphabet if it is already available registered in lower case if not it considers the upper case alphabet as a new one and increases the frequency whenever it registers the same letter either in upper case or lower case. Symbols and spaces are being considered as unique characters and the frequency of those symbols is also calculated so that we don’t lose any data while encoding.

**Submitted by:**

Pragnya Gannapureddy

Nitheesh Edla