Huffman Compression & Decompression

Huffman Coding is a famous Greedy Algorithm. It is used for the lossless compression of data. It uses variable length encoding. It assigns variable length code to all the characters. The code length of a character depends on how frequently it occurs in the given text. The character which occurs most frequently gets the smallest code. The character which occurs least frequently gets the largest code.

Huffman Coding implements a rule known as a prefix rule. This is to prevent the ambiguities while decoding. It ensures that the code assigned to any character is not a prefix of the code assigned to any other character.

Steps in building Huffman tree

Step 1

- Create a leaf node for each character of the text.
- Leaf node of a character contains the occurring frequency of that character.

Step 2

- Arrange all the nodes in increasing order of their frequency value.

Step 3

- Considering the first two nodes having minimum frequency,
- Create a new internal node.
- The frequency of this new node is the sum of frequency of those two nodes.
- Make the first node as a left child and the other node as a right child of the newly created node.

Step 4

- Keep repeating Step 2 and Step 3 until all the nodes form a single tree.
- The tree finally obtained is the desired Huffman Tree.

Step 5

- Assign 0 to each left child and assign 1 to each right child in the Huffman tree to get the unique code for each unique character (leaf node).
- The unique codes are used to compress the file.

Data Structures

- Map to store ascii character as key and its frequency as value
- Map to store code as key and its corresponding ascii char to decode the compressed file.
- Min Heap to build the Huffman tree to get the codes.

Time complexity

- O(nlogn) where n is the number of unique characters and number of initial nodes.
- If there are n nodes, extractMin() is called 2*(n – 1) times. extractMin() takes O(logn) to extract min from min heap. So, overall complexity is O(nlogn).

Header Format

- Number of distinct characters
- Character(no space)Corresponding Code
- Number of padded zeros