GE103

HUFFMAN CODING

2021EEB1206 – Ritika Dashore

INTRODUCTION TO HUFFMAN CODING –Huffman coding is a compression technique used to reduce the size of the data or a message. It follows an algorithm to store data in a reduced form so that when the data is sent over a network, it can be compressed to reduce the cost of transmission. Every data item is assigned a variable length of prefix code (variable length code in which no code is a prefix of another). Huffman coding is optimal because the Huffman tree gives us the codes that are unique and can hence it can be uniquely decodable code. The most frequent character is assigned the smallest code and the least frequent character is assigned the largest code.

Huffman's Greedy Algorithm uses a table of frequencies of occurrences of every character to construct an optimal way of representing these characters as a binary tree. A Greedy Algorithm is an algorithmic strategy that leads to the best optimal choice at every stage with the goal of eventually leading to a globally optimal solution. This means that the algorithm picks the best solution at the moment without regard for consequence.

Huffman coding was invented as an idea to construct an algorithm for the most efficient binary codes. David A. Huffman, Huffman's code inverter came up with the idea of using a frequency-sorted binary tree. This idea soon proved to be one of the greatest and most efficient algorithms for binary coding.

KEYWORDS: Huffman coding, Greedy Algorithm, Huffman tree, Encoding, Decoding.

BASIC ALGORITHM OF HUFFMAN CODING: GREEDY METHOD:

Step 1: COLLECT FREQUENCIES

1) First of all, build a frequency dictionary.
2) Then, make a sorted arrangement of the characters based on those frequencies.

Step 2: TREE BUILDING AND ASSIGNING CODES

1) Make a Huffman tree by selecting two minimum nodes and merging them.
2) Then transverse the tree from the root and assign the codes to the characters.
3) Store the above data in a file.

Step 3: ENCODING(Compress)

1) By using the above file, replace each character with its code and store it in a new file.
2) Overall size of the new file will be less than the original file.

Step 4: DECODING(Decompress)

1) Start reading the bits of the encoded file.
2) Replace the respective bit sequence (valid code) with its character.

ALGORITHM OF BUILDING HUFFMAN TREE:

1) Take the sample input data.
2) Find the frequency of each character.
3) Arrange the characters in a sorted manner so that the least value code is in the front.
4) Abstract two minimum nodes from the sorted arrangement.
5) Merge these nodes in order to make a new node
6) Make this new code the parent of the above two abstracted codes.
7) Now consider the new node into the sorted arrangement mentioned above.
8) Repeat the above steps until only a single node is left.

METHOD TO ASSIGN CODES USING HUFFMAN TREE

1) Assign 0 to all the left edges and 1 to the right edges of the Huffman tree.
2) Transverse the tree from root to the character and form codes for a particular character.

LITERATURE REVIEW:

In 1952, David A. Huffman, Student of MIT discovered this algorithm. He came up with the idea of using a frequency sorted binary tree which is able to find the most efficient binary code. He built the tree from the bottom-up instead of from the top down. There are a lot of compression

algorithms but we focus on lossless compression. Hence, the Huffman algorithm is so cool and efficient in the sort of smallest application. It is used for image compression PNG, JPG, MPEG, and for text compression for static and dynamic Huffman coding techniques.

OBJECTIVE

Nowadays, computer science has become a basic necessity in all fields and it involves storing information in the form of data on a huge scale. There are tons of ways to store information in CS. Scientists are searching for better ways to store data in as much small space as possible to reduce the cost of data transmission and storage to increase commercial and economic feasibility. All this can be achieved by reducing the bit size of data in the field of binary coding by using a smaller length code for the more frequent character and larger length codes for less frequent characters, which is not the case in UTF-8 encoding which uses ASCII codes (American Standard Code for Information Interchange) to assign the code to each character whose length is a nearest greater multiple of 8. In today's era, the algorithm that is very close to the above idea is Huffman compression.

CONCLUSIONS

As Huffman code compresses the size of a file, it is possible to store a larger amount of data in small storage. Huffman coding is helpful in data compression for frequently occurring characters It is a technique of compressing data to its reduced form without losing any detail which provides us with precise and quality information that saves both the cost and the time of the user and that is what we want to achieve.

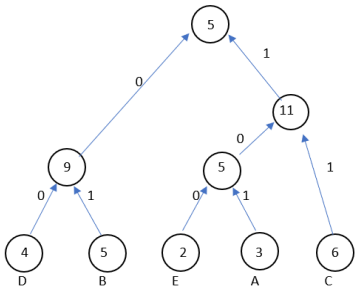
Test Cases and boundary condition:

Let there be a text message = ABAACCEBDCBCCCBEDDBD

Length of the message = 20

ENCODING:

CHARACTERS	COUNT	BITS CODE
A	3	101
B	5	01
C	6	11
D	4	00
E	2	100
	TOTAL - 20	



HUFFMAN CODE- 10101101101111110001001101111111011000000100

DECODING:

HUFFMAN CODE-10101101101111110001001101111111011000000100

BITS CODE	CHARACTER
101	A
01	B
11	C
00	D
100	E

TEXT: ABAACCEBDCBCCCBEDDBD

ACKNOWLEDGEMENT: We take this prestigious opportunity to express a deep sense of gratitude towards my project mentor, Shahid Shafi Dar sir for providing excellent guidance, encouragement, and inspiration throughout the project work. Without his guidance, this project would never have been a successful one. I would also like to thank our course instructor S. R. S. Iyengar sir for your support throughout the semester. We have put our sincere efforts into this project.

Team Pravah

REFERENCES:

https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/

Huffman coding - Wikipedia