DATA STRUCTURES AND ALGORITHMS

DR SAMABIA TEHSIN

BS (AI)

Data Compression

Data compression is the process of encoding, restructuring or otherwise modifying data in order to reduce its size. Fundamentally, it involves re-encoding information using fewer bits than the original representation.

Data Compression

Compression is often broken down into two major forms, “lossy” and “lossless”. When choosing between the two methods, it is important to understand their strengths and weaknesses:

- **Lossless Compression:** Removes bits by locating and removing statistical redundancies. Because of this technique, no information is actually removed. Lossless compression will often have a smaller compression ratio, with the benefit of not losing any data in the file. This is often very important when needing to maintain absolute quality, as with database information or professional media files. Formats such as FLAC and PNG offer lossless compression options.
- **Lossy Compression:** Lowers size by deleting unnecessary information, and reducing the complexity of existing information. Lossy compression can achieve much higher compression ratios, at the cost of possible degradation of file quality. JPEG offers lossy compression options, and MP3 is based on lossy compression.

Data Compression Uses

Most businesses today rely on data compression in some major way, especially as the functional quality of data increases, storage capacity concerns have to be resolved. Data compression is one of the major tools that helps with this. There a number of file types that are frequently compressed:

- **Audio Compression**
- **Video Compression**
- Text Compression

Techniques of data compression

Lossless:

- Lempel–Ziv (LZ)
- run-length encoding
- Huffman algorithm
- etc

Lossy:

- Transform Coding.
- Discrete Cosine Transform.
- Discrete Wavelet Transform.
- Fractal Compression.

etc

Huffman Coding

Huffman Coding is a technique of compressing data to reduce its size without losing any of the details. It was first developed by David Huffman.

Huffman Coding is generally useful to compress the data in which there are frequently occurring characters.

How Huffman Coding works?

Initial string

Each character occupies 8 bits. There are a total of 15 characters in the above string. Thus, a total of 8 * 15 = 120 bits are required to send this string.

How Huffman Coding works?

Huffman coding first creates a tree using the frequencies of the character and then generates code for each character.

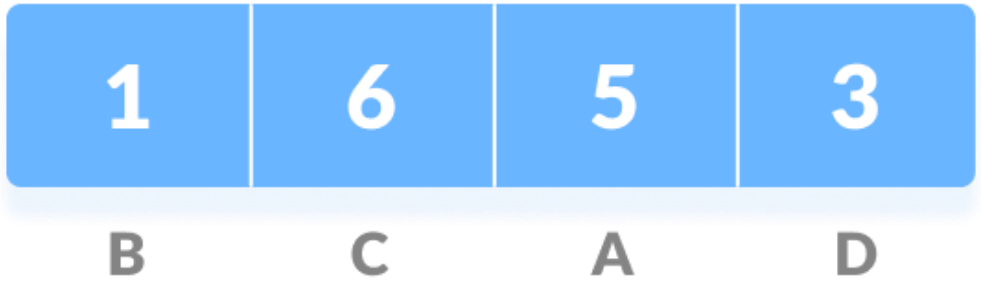
Once the data is encoded, it has to be decoded. Decoding is done using the same tree.

Huffman Coding prevents any ambiguity in the decoding process using the concept of **prefix code** ie. a code associated with a character should not be present in the prefix of any other code. The tree created above helps in maintaining the property.

How Huffman Coding works?

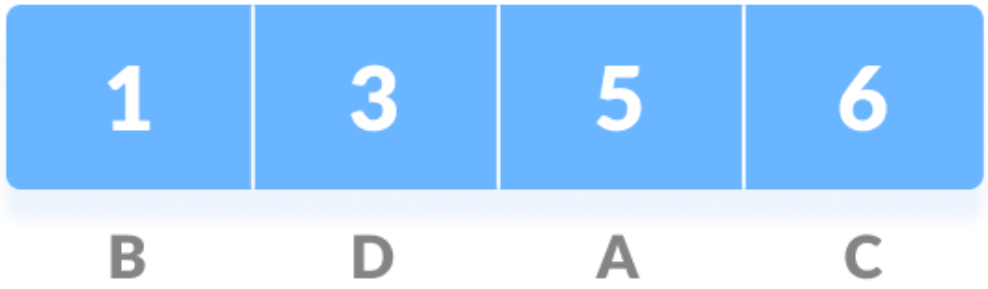
Huffman coding is done with the help of the following steps.

1.Calculate the frequency of each character in the string.



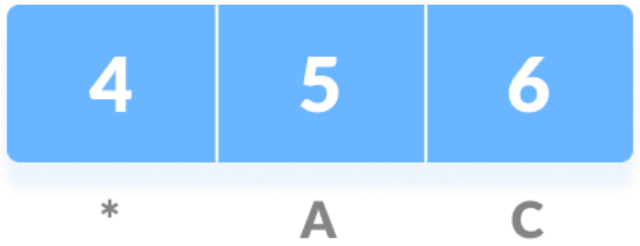
How Huffman Coding works?

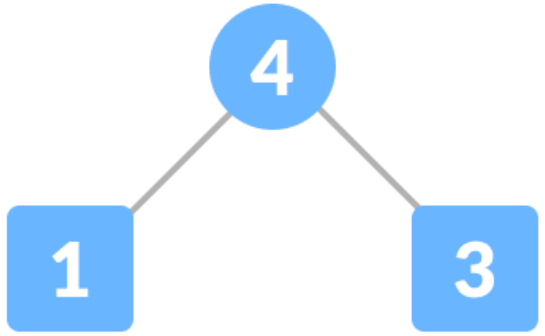
2. Sort the characters in increasing order of the frequency. These are stored in a priority queue Q.



How Huffman Coding works?

3. Make each unique character as a leaf node.
4. Create an empty node z. Assign the minimum frequency to the left child of z and assign the second minimum frequency to the right child of z. Set the value of the z as the sum of the above two minimum frequencies.



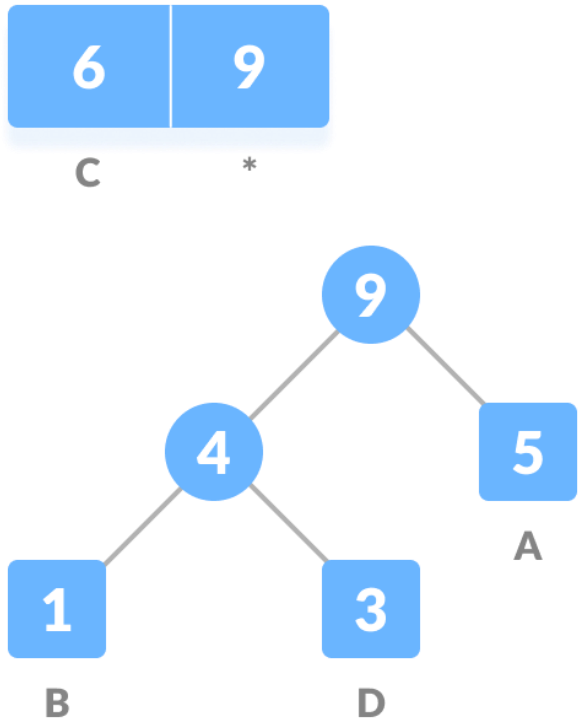


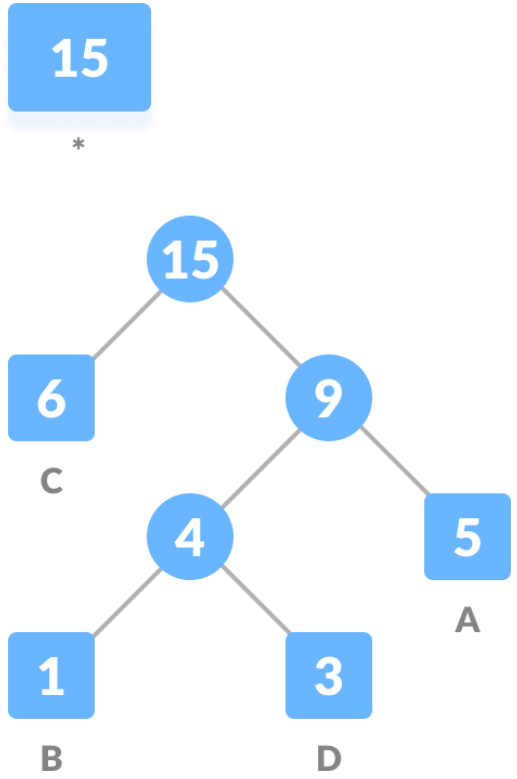
How Huffman Coding works?

5. Remove these two minimum frequencies from Q and add the sum into the list of frequencies (* denote the internal nodes in the figure above).

6. Insert node z into the tree.

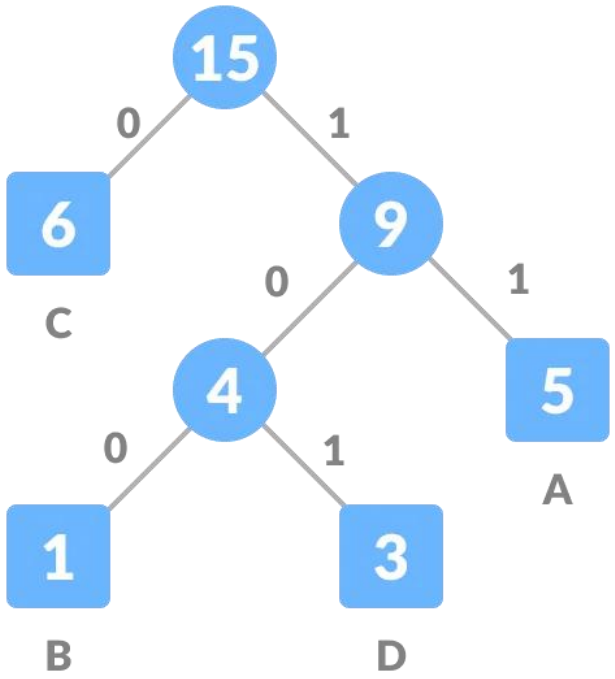
7. Repeat steps 3 to 5 for all the characters.





How Huffman Coding works?

8. For each non-leaf node, assign 0 to the left edge and 1 to the right edge



The total size is given by the table below.

Character	Frequency	Code	Size
A	5	11	5*2 = 10
B	1	100	1*3 = 3
C	6	0	6*1 = 6
D	3	101	3*3 = 9
4 * 8 = 32 bits	15 bits		28 bits

Compression Ratio

Data compression ratio, also known as **compression power**, is a measurement of the relative reduction in size of data representation produced by a data compression algorithm. It is typically expressed as the division of uncompressed size by compressed size.

$$\text{Compression Ratio} = \frac{\text{Uncompressed Size}}{\text{Compressed Size}}$$

Credits and Acknowledgements

https://www.gatevidyalay.com

https://www.programiz.com/