Compression:

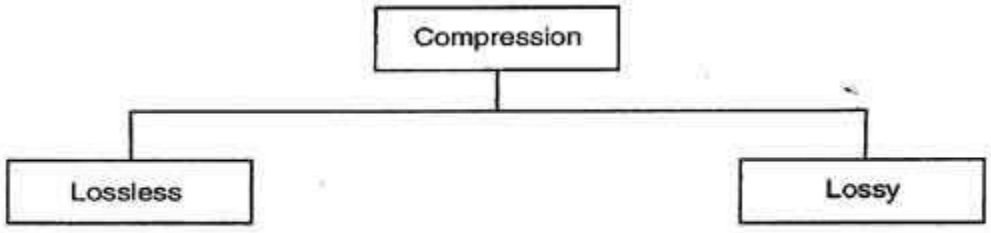
Data compression is the function of presentation layer in OSI reference model. Compression is often used to maximize the use of bandwidth across a network or to optimize disk space when saving data.

There are two general types of compression algorithms:

1. Lossless compression

2. Lossy compression

Types of Compression



Types of Compression

Lossless Compression

Lossless compression compresses the data in such a way that when data is decompressed it is exactly the same as it was before compression i.e. there is no loss of data.

A lossless compression is used to compress file data such as executable code, text files, and numeric data, because programs that process such file data cannot tolerate mistakes in the data.

Lossless compression will typically not compress file as much as lossy compression techniques and may take more processing power to accomplish the compression.

Lossless Compression Algorithms

The various algorithms used to implement lossless data compression are :

1. Run length encoding
2. Differential pulse code modulation
3. Dictionary based encoding

1. Run length encoding

• This method replaces the consecutive occurrences of a given symbol with only one copy of the symbol along with a count of how many times that symbol occurs. Hence the names ‘run length'.

• For example, the string AAABBCDDDD would be encoded as 3A2BIC4D.

• A real life example where run-length encoding is quite effective is the fax machine. Most faxes are white sheets with the occasional black text. So, a run-length encoding scheme can take each line and transmit a code for while then the number of pixels, then the code for black and the number of pixels and so on.

• This method of compression must be used carefully. If there is not a lot of repetition in the data then it is possible the run length encoding scheme would actually increase the size of a file.

3. Dictionary based encoding

• One of the best known dictionary based encoding algorithms is Lempel-Ziv (LZ) compression algorithm.
• This method is also known as substitution coder.
• In this method, a dictionary (table) of variable length strings (common phrases) is built.
• This dictionary contains almost every string that is expected to occur in data.
• When any of these strings occur in the data, then they are replaced with the corresponding index to the dictionary.
• In this method, instead of working with individual characters in text data, we treat each word as a string and output the index in the dictionary for that word.

• For example, let us say that the word "compression" has the index 4978 in one particular dictionary; it is the 4978th word is usr/share/dict/words. To compress a body of text, each time the string "compression" appears, it would be replaced by 4978.

Lossy Compression

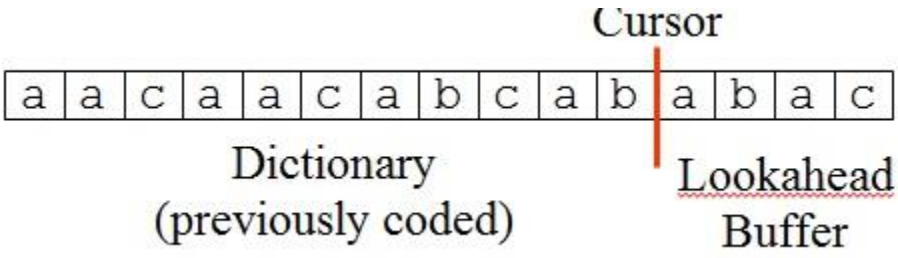
Lossy compression is the one that does not promise that the data received is exactly the same as data send i.e. the data may be lost.
This is because a lossy algorithm removes information that it cannot later restore.
Lossy algorithms are used to compress still images, video and audio.
Lossy algorithms typically achieve much better compression ratios than the lossless algorithms.
The **Lossy compression** method eliminates some amount of data that is not noticeable.

Compression Ratio
Data compression ratio is defined as the ratio between the uncompressed size and compressed size.

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

Thus, a representation that compresses a file's storage size from 10 MB to 2 MB has a compression ratio of 10/2 = 5.

LZ77: Sliding Window Lempel-Ziv



Dictionary and buffer "windows" are fixed length and slide with the cursor
Repeat:
Output (*p*, *l*, c) where
p = position of the longest match that starts in the dictionary (relative to the cursor)
l = length of longest match
c = next char in buffer beyond longest match
Advance window by *l* + 1