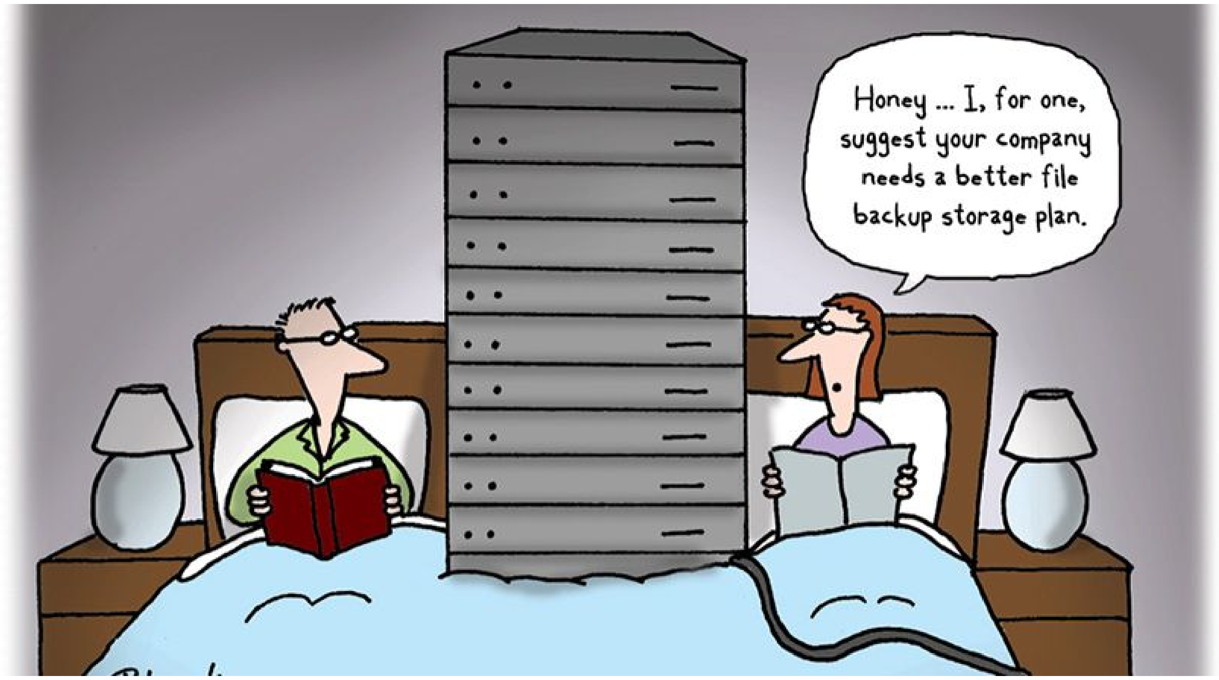
Algorithm Design and Analysis Lab : CS254

LZW(Lempel–Ziv–Welch) Compression Algorithm

Arushi Jain (160001008)

Naman Singhal (160001038)


Honey ... I, for one, suggest your company needs a better file backup storage plan.

Introduction

Data compression is the algorithmic problem of finding alternative, space-efficient encodings for a given data file. With the rise of computer networks, a new mission for data compression has arisen, that of increasing the effective bandwidth of networks by reducing the number of bits before transmission.

Lossless compression is a class of data compression algorithms that allows the original data to be perfectly reconstructed from the compressed data. This is different from lossy compression where some loss of information is acceptable. Here we are considering one of the efficient lossless compression algorithm, LZW.

LZW Compression

LZW compression works by reading a sequence of symbols, grouping the symbols into strings, and converting the strings into codes. Because the codes take up less space than the strings they replace, we get compression. It is an improvement on other lossless compression algorithms like LZ78 and Huffman Algorithm. This algorithm is typically used in GIF and optionally in PDF and TIFF, Unix's 'compress' command, among other uses.

Objectives

The project will initially focus on design and implementation of the LZW Compression and Decompression Algorithms. This will be followed by the efficiency analysis of the algorithms in terms of the tradeoff between the compression ratio and the processing time. The project will then aim at optimizing the basic algorithm and finally conclude with a comparison with other lossless data compression techniques.

1. LZW Compression Algorithm
    a. Design
    b. Implementation
    c. Efficiency Analysis
        i. Compression Ratio
        ii. Compression Time
2. LZW Decompression Algorithm
    a. Design
    b. Implementation
    c. Efficiency Analysis
        i. Decompression Time
3. Improvising LZW using Dynamic Bit Sizes and Optimized Dictionary
4. Comparison With LZ78 and Basic Huffman Techniques