Computer Engineering Dept.
Cairo university

Multimedia Project Report

Prepared By :

Team 1

- **Mohammed Magdy**
- **Hassan Osama**
- **Muhammad AbdulKariim**
- **Hussein Youssef**

History of Experiments

We first tried Huffman algorithm and managed to get a reasonable compression ratio, We then searched for other algorithms like canonical Huffman and LZSS, We chose to implement LZ77 to use it in Deflate algorithm but it didn't work out. We finally implemented LZW and did some modifications to it.

Final Algorithm

We finally decided to implement LZW with some modifications. We prepared a dictionary of all possible symbols to use it. We use specific number of bits for each number and output it as the first byte in the binary file then we apply the algorithm and get an array of integers we then convert these integers to binary then to bytes to output it in the binary file. In Decoding we build the dictionary first then we read the bytes from the binary file byte by byte. The first byte will represent the number of bits for each number that was used in encoding. We then convert the bytes to binary then to integers and use the dictionary to get back the characters. The encoder needs the path to the dataset and dictionary file. The decoder needs the path to the compressed binary file and the dictionary file.

Content of Compressed file

The compressed file has a byte (the first byte) which represents the number of bits used to represent each integer. The rest of the bytes are the codewords for the strings so we can recover the string using the dictionary.

Files sent to The Decoder

The Decoder needs only the compressed binary file and the file that the encoder used to build the dictionary and it'll operate as it should.

Results

- Dataset 1:
Bits = 531557 * 1024 bits
Compression Ratio = 3.75

- Dataset 2:
Bits = 468436 * 1024 bits.
Compression Ratio = 3.76

- Dataset 3:
Bits = 369705 * 1024 bits
Compression Ratio = 3.76

- Dataset 4:
Bits = 496829 * 1024 Bits
Compression Ratio = 3.72

- Dataset 5:
Bits = 576882 * 1024 Bits
Compression Ratio = 3.77

- Dataset 6:
Bits = 470287 * 1024 Bits
Compression Ratio = 3.8

- Dataset 7:
Bits = 520176 * 1024 Bits
Compression Ratio = 3.71

- Dataset 8:
Bits = 481669 * 1024 Bits
Compression Ratio = 3.73

- Dataset 9:
Bits = 542712 * 1024 Bits
Compression Ratio = 3.83

- Dataset 10:
Bits = 593820 * 1024 Bits
Compression Ratio = 3.8

- Dataset 11:
Bits = 522904 * 1024 Bits
Compression Ratio = 3.76

- Dataset 12:
Bits = 502721* 1024 Bits
Compression Ratio = 3.67

- Dataset 13:
Bits = 530272 * 1024 Bits
Compression Ratio = 3.74

- Dataset 14:
Bits = 508809 * 1024 Bits
Compression Ratio = 3.69

- Dataset 15:
Bits = 505766 * 1024 Bits
Compression Ratio = 3.75

- Dataset 16:
Bits = 475413 * 1024 Bits
Compression Ratio = 3.71

- Dataset 17:
Bits = 505541 * 1024 Bits
Compression Ratio = 3.71

- Dataset 18:
Bits = 545578 * 1024 Bits
Compression Ratio = 3.73

- Dataset 19:
Bits = 502868 * 1024 Bits
Compression Ratio = 3.73

- Dataset 20:
Bits = 517048 * 1024 Bits
Compression Ratio = 3.73

Average Compression Ratio = 3.75
Average Number of bits = 508450 * 1024 Bits

Results of Previous algorithms

We could get average Compression Ratio = 3.2 using Huffman algorithm and 2.0 using LZ77 algorithm.

Work Division:

1- Mohammed Magdy : Searching and Gathering information.
2- Hussein Youssef : Implementing LZW Encoder and LZ77.
3- Hassan Osama : Implementing LZW Decoder.
4- Muhammad Abdulkariim : Implementing Huffman.