TOPIC- TEXT FILE COMPRESSOR

-------------------------------------------

ADITYA RAJ<E22CSEU0649>

SANJANA NAYAK<E22CSEU0644>

INTRODUCTION

File compression is a critical technique for reducing the size of one or more files, optimizing storage space, and enhancing file transfer speeds across networks. In our project, we aim to develop a Text File Compressor Tool that utilizes ten different compression algorithms. We will evaluate and compare the performance of these algorithms by measuring the time taken for compression and the resulting compressed file sizes.

LITERATURE REVIEWS OF EXISTING METHODOLOGY

1)Huffman Coding

* Huffman coding is a widely used algorithm for lossless data compression.
* It is a variable-length prefix coding technique, where frequently occurring characters or symbols are assigned shorter codes.
* Huffman coding is used in popular compression formats like ZIP and DEFLATE.

2)Lempel-Ziv-Welch

* LZW is another well-known lossless compression algorithm.
* It is used in formats like GIF and the UNIX compress utility.
* LZW works by building a dictionary of recurring patterns and replacing them with shorter codes.

3) Burrows-Wheeler Transform (BWT):

* The BWT is a reversible permutation-based transformation technique.
* It is often used as a preprocessing step in conjunction with other compression algorithms like the Move-to-Front (MTF) transform in the BZIP2 compression format.

4) Delta Encoding:

* Delta encoding is used for compressing data where adjacent values are similar.
* It stores the difference between successive data points rather than the absolute values.

5) Run-Length Encoding (RLE):

* RLE is a simple lossless compression technique suitable for compressing data with long runs of repeated characters or symbols.
* It replaces consecutive occurrences of a character with a count of the repeated character and the character itself.

REASEACRH GAP

Here are some points in which research has been going on:

1)Improved Compression Algorithms:

* Develop more efficient and effective compression algorithms that can achieve higher compression ratios while maintaining fast compression and decompression speeds.
* Explore new techniques that leverage machine learning and artificial intelligence for data compression.

2) Security and Privacy in Compression:

* Investigate the security implications of compression, including potential vulnerabilities related to compressed data.
* Develop compression techniques that can ensure data privacy and security, especially in applications involving sensitive information.

3) Compression for Large-Scale Data:

* Address the challenges of compressing and decompressing large datasets, including distributed compression methods for big data applications.
* Research scalable compression techniques suitable for cloud computing environments.

PROPOSED METHODOLOGY

**PROPOSED METHODOLOGY:**

**Project Scope:** Our project focuses on developing a Text File Compressor Tool that employs ten distinct compression algorithms. The tool will compress text files and provide detailed information about each compression, including the resulting compressed file, the time taken for compression, and the compressed file's size for each algorithm.

**Data Preprocessing:** Before compression, we will implement data preprocessing steps to enhance the efficiency and quality of compression:

1. **Data Preparation:** Ensure the input text files are ready for compression, and any required adjustments are made.
2. **Feature Extraction:** Identify relevant features or patterns within the text data that can aid in compression.
3. **Data Cleansing:** Remove unnecessary elements, whitespace, or noise from the text data.

**Compression Algorithms:** We will implement ten different compression algorithms, each with its unique approach to text file compression. These algorithms will be selected based on their ability to achieve efficient compression while meeting the project's goals.

**Measurement and Evaluation:** For each compression using the ten algorithms, we will:

1. Record the compressed file generated.
2. Measure the time taken for compression.
3. Determine the size of the compressed file.

**Comparative Analysis:** We will analyze and compare the performance of the ten compression algorithms based on the collected data. This analysis will help identify which algorithms are most effective in terms of compression ratio and computational efficiency.

**Conclusion:** Our Text File Compressor Tool will provide valuable insights into the performance of various compression algorithms when applied to text files. This information will assist users in selecting the most suitable compression method based on their specific requirements, ultimately contributing to more efficient text file management.

DETAILS OG TEAM CONTRIBUTION

We have divided the work into half. We took 14 Algorithm and decided to do 7 -7 and use only 10 working Algorithms.

So, **Aditya has (RLE, LZE, BWT, Arithmetic Coding, Bzip2, LZ77 and LZ78 and PPM)** and

**Sanjana has (Delta Encoding, BWT, Deflate Compression, Shannon Fano Coding, AHC, LZMA, SBV)** to code.

And later compression and implementation we would try to do it together.