Nicolas Corfmat
CSE13S, Winter 2023
ncorfmat@ucsc.edu

Maximizing Efficiency Using Data Compression

Assignment 6: Lempel-Ziv Compression

1 Description of Program

This project consists of two main programs: *encode* and *decode*, which implement LZ78 compression and decompression respectively. The *encode* program is used to compress either text or binary files while *decode* decompresses compressed files.

2 The Utility of Data Compression

In the digital world, data is constantly being sent and received. Whether you are streaming a Netflix movie or uploading a post to Instagram, the reality is that all of these activities require binary information to be physically stored somewhere. Now imagine how inefficient it would be if Instagram were to store thousands of copies of the same picture, or if you were trying to stream movies in 4K resolution with slow internet speeds. In both cases, it would require more money to be spent on more capable equipment. At this point, it becomes clear why someone would want to minimize the amount of data being exchanged.

Fortunately, there exist a myriad of compression algorithms that reduce unnecessary information and help maximize storage. Reducing file sizes has the advantage of lowering transmission times as a result of a decrease in information that needs to be uploaded and downloaded. The two man types of data compression are known as *lossless* and *lossy* compression.

Lossless compression refers to compression algorithms that reduce file size without erasing any data, so when the file is decompressed, it returns to its original state. This method works by replacing repeating sequences with a single code.

Lossy compression refers to size reduction that erases unnecessary data. For example, a file representing a picture of a sunny blue sky would hold a vast amount of information surrounding pixel location and color. It would be quite redundant storing millions of pixels representing roughly the same shade of blue, at which point, lossy compression becomes helpful.

3 The Program

For this assignment, we will be using *Lempel-Ziv Compression*, otherwise known as LZ78, to perform lossless compression on text and binary files. It works by creating dictionaries that store character patterns and a corresponding reference to its location in the dictionary. With the help of the tree-like data structures known as *Tries*, we can represent this dictionary along with its symbols and references.

A Trie works by establishing a parent node that contains a symbol and n child nodes. Depending on which alphabet you are using, n can vary, but for this program, we will be using the ASCII characters ($n = 256$). Every time a symbol is analyzed, the program checks whether it has been assigned a location within the parent node. If the answer is yes, it looks at the next character and determines if it needs to create a new child or recognize that is has already been referenced. This cycle continues until there is no more data left to read. With this approach, repeating words in a file such as “hello” and “hello” for example, would be assigned a location within the dictionary once and ultimately have the effect of reducing the file’s size.

4 Limitations to Lempel-Ziv Compression

Lempel-Ziv Compression is not perfect. Much like any solution to a problem, there are trade offs to choosing one approach over the others. Since LZ78 works to eliminate redundancies, it

is not effective at compressing files with minimal to no patterns. For example, a text file containing the entire A-Z alphabet would be unchanged under this method of compression compared to a file consisting of only the letter "A". In programming, this measure of randomness is referred to as *entropy*, and it determines how effective compression is at reducing unnecessary data.

Moreover, since LZ78 utilizes a tree-like node structure, it becomes a hassle decompressing certain sections without first decompressing the previous data. Since the nodes on the Tries are all connected, you must begin at the root to access a particular sequence of characters. Essentially, its ability for random access is limited.

5 Conclusion

Data compression is an effective approach to reducing file sizes and maximizing the limited space of computers. The two main types of compression approaches—lossy and lossless—allow people to choose between irreversibly shrinking files by deleting redundancies and minimizing size by identifying repeating patterns respectively. Lempel-Ziv compression reduces file sizes by adopting the lossless approach, yet it becomes increasingly limited as entropy, or the randomness of the data, increases. Ultimately, data compression is essential to creating efficient and high-performing digital infrastructure which explains why it is so common today.