Asgn5 Design Doc

Elliott Jensen

October 2021

1 Introduction

This program is a Huffman coding algorithm that features an encode program that can reduce the file size of the user input. Rather than mapping every character to an 8-bit code, the program assigns the most common characters smaller codes and rarer characters longer codes in order to losslessly reduce file size. It also features a decoding algorithm which can take the header produced by the encoding algorithm and produce the original text.

2 Encoding: encoding.c

This file includes the first call to main and uses getopt to allow the user to specify an input and output file along with specifying if the user would like the statistics of the data compression to be printed out.

Essentially this file: 1. Generates a histogram - maps each character to its frequency 2. Construct a Huffman tree - uses a priority queue to order nodes 3. Construct a code table 4. Emit an output header file

Most importantly, this file uses low level system calls (read, write, open and close) located in io.c in order to receive information from the input file that is then used to create the histogram and Huffman tree. In order to create the histogram I will use the readbytes function in io.c:

```
for i in  ALPHABET:
    hist[i] = 0
while bytes_read = read_bytes(infile,buf,BLOCK) > 0:
    hist[buf[i]] += 1
```

I will then place each item in the histogram into a priority queue so that the items with the highest frequency are given the highest priority. I will then construct a heap where I add the items of highest priority first which will create the Huffman tree. To do this I will rely on three files which each contain their own data structure and respective functions: stack.c, node.c and pq.c. Following the pattern of left meaning 0 and right meaning 1. I will use the Huffman tree to construct a code table. I will then write this, along with the the encoding of the Huffman tree into the header file which I will then output and which can be processed by decode.

3 Decoding: decoding.c

This file uses the code table to read the binary stream outputted by the encoding file. It decompresses the file by traversing down the Huffman tree that is reconstructed using the header file.