ELEN-0060: Information and Coding Theory

Project 2 - Source coding, data compression and channel coding

Maxime Goffart
180521

Olivier Joris
182113

Academic year 2021 - 2022

1 Source coding and reversible data compression

1.1 Question 5

First, we can compute the marginal probability distribution of all the symbols based on the given Morse text. The distribution is:

Symbol	.	-	_	/
Probability	0.43378	0.28706	0.21452	0.06464

Table 1: Marginal probability distribution of all the symbols

Based on the distribution of probabilities, we can compute the binary Huffman code. We get the following code:

Symbol	.	-	_	/
Huffman code	0	11	101	100

Table 2: Binary Huffman code

By applying the obtained Huffman code to the Morse text, we get the encoded Morse text whose size is 2213141 bits. The original Morse text has a size of 2398580 bits. Thus, we have a compression rate of 1.08379.

1.2 Question 6

We can compute the expected average length of the Huffman code by computing the sum for the 4 symbols of the probability of each symbol times the length of the code associated with the symbol. We get that the expected average length is equal to 1.84538.
By comparing to the empirical average length of the Huffman code, we get:

$$\text{Length of encoded / length of initial text} = 1.84538$$

This value is the same as the one for the expected average length which is logical since the expected average length was computed based on the probabilities of occurrence of each symbol based on the given Morse text.
By comparing to the theoretical bounds, we get that:

$$\frac{H(S)}{log_2(q)} \leq \overline{n} \leq \frac{H(S)}{log_2(q)} + 1 \text{ because } 1.77138 \leq 1.84538 \leq 2.77138 \tag{1}$$

Thus, our code is optimal, because the inequation is satisfied, which was expected because Huffman codes are optimal. But, the obtained code is not absolutely optimal because $\frac{H(S)}{log_2(q)} \neq \overline{n}$.

1.3 Question 7

For the evolution of the empirical average length of the Huffman code with respect to the lengths of the input texts, we get the following plot:

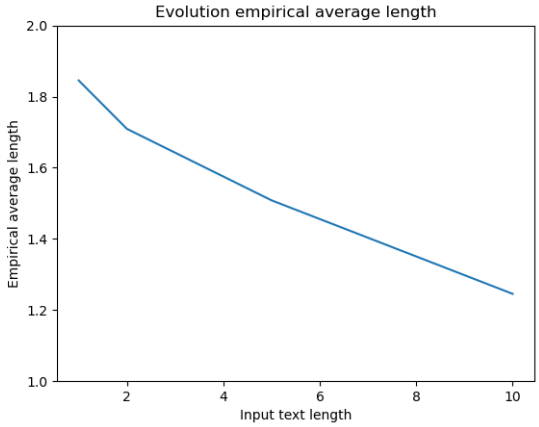

Figure 1: Evolution of empirical average length

As we can observe on the plot, the empirical average length decreases as the length of the input text increases.

1.4 Question 14

By computing the Huffman code on the original (27 symbols) text, we get the following code:

Symbol	**a**	**b**	**c**	**d**	**e**	**f**	**g**
Code	1010	100000	100010	11010	001	100011	00000
Symbol	**h**	**i**	**j**	**k**	**l**	**m**	**n**
Code	0111	0100	1100111001	1100110	11000	110010	0110
Symbol	**o**	**p**	**q**	**r**	**s**	**t**	**u**
Code	1001	100001	1100111011	0101	0001	1011	00001
Symbol	**v**	**w**	**x**	**y**	**z**		
Code	11001111	110110	1100111010	110111	1100111000	111	

Table 3: Binary Huffman code for original text

The expected average length of the code is 4.15047 and the experimental length of the encoded text is 1711279 bits.
The original text has a size of 2061550 bits and the encoded text has a size of 1711279 bits. Thus, the compression rate is 1.2046 (= 2061550/1711279).

1.5 Question 15

By comparing the values obtained at the questions 5 and 14, we can conclude that it is better to encode directly the text without first converting it to Morse because the size of the encoded Morse text is 2213141 bits while the size of the encoded text is 1711279 bits. Furthermore, the compression rate obtained is higher in question 14 compared to question 5.
This can be justified theoretically by the fact that using Huffman encoding with 27 symbols instead of 4 symbols allows to benefit more from the frequencies of letters in English. For instance, the letter z in English is way less frequent than the letter e. While using the Morse encoding, the 2 symbols . and - both appear for some of the letters. Thus, we loose the benefit provided by less frequent letters that will get longer codes and more frequent letters will get shorter codes.