Introduction to Information Theory, Fall 2019

Homework problem set #3 — due September 27, 2019

Rules: Always explain your solutions carefully. You can work in groups, but must write up your solutions alone. You must submit your solutions before the Friday exercise class (either in person or by email).

1. **Human compressor (1 point):** Consider the following probability distribution:

x	A	B	C	D	E	F	G
P(x)	0.05	0.05	0.07	0.13	0.2	0.2	0.3

   (a) Construct a Huffman code C for P and compute the average length per symbol $L(C, P)$.

   Now imagine running the Lempel-Ziv (LZ) algorithm on the following string:

   `XOOXOXOOXOXXXXOOOOX⊥`

   (b) List the distinct phrases that the LZ algorithm splits the string into.

   (c) List the pairs (k, x) generated by the LZ algorithm.
   Hint: Write k as an integer in decimal notation and $x \in \{\mathtt{X}, \mathtt{O}, \perp\}$.

2. **Kraft-McMillan inequality (1 point):** Let X be a random variable with distribution P. In class, we discussed that there always exists a prefix code C whose codewords have length $\ell(C(x)) = \lceil \log 1/P(x) \rceil$. Let C' be any other uniquely decodable code. Show that

$$\Pr\big(\ell(C(X)) \geqslant \ell(C'(X)) + k\big) \leqslant \frac{1}{2^{k-1}}$$

   for all k. This means that no other code can produce significantly shorter codewords than C most of the time.

   Hint: Use the Kraft-McMillan inequality.

3. 🖩 **Compression algorithms (1 point):**
   In this problem, you will implement the Huffman and Lempel-Ziv algorithms discussed in class and above. To get started, open the notebook at https://colab.research.google.com/github/amsqi/iit19-homework/blob/master/03-homework.ipynb and follow the instructions.

   As always, please submit your solution as a Python notebook or script, or as a PDF printout. You can score the maximum score if your solution produces the correct output. We will only have a closer look at your code in case of problems.

   This programming problem may be a bit more difficult than last week's, so we will grade it gently. We also added some optional challenge problems in the notebook (upon popular request). Can you beat `zlib` *at compressing Wikipedia?* ☺