EE 231A — Handout #7, Assessment for Part 2, Spring 2020
Information Theory — Wednesday, April 29, 2020
Instructor: Rick Wesel — Due Tuesday May 12, 2020 before class

This assessment is open book and open note, but you may not perform an internet search to seek a worked solution to a problem. Do not ask for help from your classmates or anyone else besides Prof. Wesel and Hengjie.

(30 pts) *Compression to the Entropy and the Entropy Rate.*
In problem set 2, problem 8 *Random walk on a chessboard*, you computed that the entropy rate for the random walk of a king on a 9-square chess board is 2.24 bits per symbol. In this problem you implement arithmetic coding and decoding to compress and recover a short sequence of this random walk. You will implement both a version of arithmetic coding that targets the *entropy* of the stationary distribution and a version of arithmetic coding that targets the *entropy rate* of the stochastic process. The fun part at the end applying your arithmetic coding algorithms to a LONG sequence to see how close you can get to the entropy and the entropy rate. Please don't worry about running your decoders successfully on the long sequence. There are some (solvable) numerical issues for decoding a long sequence that we won't make you explore in this assignment.

1. (0 pts) Generate your personalized long and short sequences of king movements.

   In order to generate your personalized two sequences. Do the following steps.

   (a) Download "ECE_231A_Assessment_2.zip" on CCLE .

   (b) Run `Sequence_Generator(s)` in MATLAB command window, where `s` is your UCLA UID number. This will generate two .txt files under your current directory named as

   "king_movements_long_sequence_X.txt"

   "king_movements_short_sequence_X.txt"

   where X is your unique sequence identifier. **Please do not tamper with the MATLAB function, the pre-generated sequences, and the location of the two folders inside.** We will use your UID to verify whether you are working on the correct two sequences.

   Data format: The long sequence contains 10^4 digits from 1 to 9, corresponding to the 9 positions on the 3×3 chessboard and a final "EOT" symbol denoted by 0 at the end. The short sequence contains 15 digits from 1 to 9 corresponding to the 9 positions on the 3×3 chessboard and a final "EOT" symbol denoted by 0 at the end. The first digit in the two sequences is generated according to the stationary distribution and the following digits (without the final EOT symbol) are generated according to the transition matrix of the Markov chain.

2. (10 pts) *Arithmetic coding that targets the* **entropy** *of the stationary distribution.*

   (a) What was the stationary distribution that you computed for the nine possible positions of the king in problem 8 of problem set 2?

   (b) Compute the entropy of this stationary distribution.

   (c) Using only the 15-digit "short" sequence, implement an arithmetic coder and the associated decoder that seeks to compress to the entropy of the stationary distribution. For the EOT symbol, use a probability of 1/15, since the EOT comes at the end of 15 symbols. You are encouraged to use `encode_symbol`, `send_bit`, `binary2real`, and `decode_symbol` from problem 10 of problem set 3 as a starting point for this part. This part is very much like problem 10 of problem set 3, only with ten intervals instead of four, so I hope it goes easily for you. For this part, turn in your MATLAB code and a diary of you running your encoder and decoder and recovering your 15 move sequence. We want to see the binary sequence produced by the encoder in the diary.

   (d) Now we will encode the LONG sequence. Adjust your probability of the EOT symbol to be 1/100. Actually the EOT symbol occurs only once in 10^4 symbols, but let's not make the probability too small. Run your encoder on the sequence of 10^4 moves. Do not report the specific binary sequence you produce, and we are explicitly NOT asking you to recover the sequence with your decoder. There are (solvable) numerical issues with using your decoder on such a long sequence that are outside the scope of this assignment. What we want you to do is to report the number of bits your decoder needed to represent the 10^4 moves. Divide that number by 10^4 and compare this to the entropy of the stationary distribution. How close did you get to the entropy in terms of bits? What was the percent of excess redundancy, computed as $100 \times \frac{\text{Rate}-H(X)}{H(X)}$?

3. (20 pts) *Arithmetic coding that targets the* **entropy rate** *of the stochastic process.*

   (a) For each of the nine positions, what is the conditional distribution for the next position of the king in problem 8 of problem set 2?

   (b) What is the entropy of the PMF or the next position given each of the nine possible starting positions?

   (c) What was the entropy rate for the stochastic process that you computed in problem 8 of problem set 2?

   (d) Using only the 15-digit "short" sequence, implement an arithmetic coder and the associated decoder that seeks to compress to the *entropy rate of the stochastic process*. For the EOT symbol, use a probability of 1/15, since the EOT comes at the end of 15 symbols. The key aspect of this encoder is that the specific interval probabilities that the arithmetic encoder uses *depends on the previous symbol.* Indeed, even the *number* of intervals depends on the previous symbol because the number of positions available to the king depends on its position. For the first symbol, use the probabilities of the stationary distribution. After that, you should use the probabilities of the conditional distribution given the current position of the king. While you can start with `encode_symbol`, `send_bit`, `binary2real`, and `decode_symbol` from the previous part, you will need to alter your encoder and decoder to be adaptive based on the previously encoded/decoded symbols. For this part, turn in your MATLAB code and a diary of you running your encoder and decoder and recovering your 15 move sequence. We want to see the binary sequence produced by the encoder in the diary.

   (e) Now we will encode the LONG sequence. Adjust your probability of the EOT symbol to be 1/100. Actually the EOT symbol occurs only once in 10^4 symbols, but let's not make the probability too small. Run your new adaptive encoder on the sequence of 10^4 moves. Do not report the specific binary sequence you produce, and we are explicitly NOT asking you to recover the sequence with your decoder. There are (solvable) numerical issues with using your decoder on such a long sequence that are outside the scope of this assignment. What we want you to do is to report the number of bits your decoder needed to represent the 10^4 moves. Divide that number by 10^4 and compare this to the entropy rate of the stochastic process. How close did you get to the entropy rate in terms of bits? What was the percent of excess redundancy, computed as $100 \times \frac{\text{Rate}-H(\mathcal{X})}{H(\mathcal{X})}$?

   (f) What was the percentage rate reduction, computed as $100 \times \frac{R_1 - R_2}{R_1}$, that was achieved by compressing to the entropy rate instead of the entropy? Here, R_1 is the rate achieved by arithmetic encoding compressing with a target of the entropy and R_2 is the rate achieved by compressing with a target of the entropy rate.