Algorithms in the Real World (15-853), Spring 13
Assignment #1 Due: Tuesday, January 29

Complete all problems.

You are not permitted to look at solutions of previous year assignments. You can work together in groups, but all solutions have to be written up individually. If you get information from sources other than the course notes and slides, please cite the information, even if from Wikipedia or another textbook.

Problem 1: Conditional Probabilities (10pt)

Given the following conditional probabilities for a two state Markov Chain what factor would one save by using the conditional entropy instead of the unconditional entropy?

$$\begin{array}{cc} p(w|w) = .92 & p(b|w) = .08 \\ p(w|b) = .2 & p(b|b) = .8 \end{array}$$

Problem 2: Arithmetic Codes (10pt)

Given the following probability model:

Letter	$p(a_i)$	$f(a_i)$
a	.1	0
b	.2	.1
c	.7	.3

Decode the 4 letter message given by `00011011010` assuming it was coded using arithmetic coding. Why is this message longer than if we simply had used a fixed-length code of 2 bits per letter, even though the entropy of the set $\{.1, .2, .7\}$ is just a little more than 1 bit per letter. Note: once you figure out how to do the decoding, it should not take more than five minutes on a calculator or scripting language.

Problem 3: Decoding Prefix Codes (20pt)

Being able to quickly decode prefix codes is extremely important in many applications.

Assume you have a machine with word length w (e.g. 32 bits). Assume you are give some prefix code for a message set, and the longest codeword is $w/2$ bits (or less). Assume that a sequence of codes is stored in memory broken into words (i.e. the first w bits are in the first word, etc.).

The naive way to decode is to use a binary tree and take constant time per bit by traversing the tree. Describe how to decode each codeword in constant time (independently of w). Hint, you can use $O(2^{w/2})$ preprocessing time, and the same amount of memory.

Please don't use more than half a page to describe the method

Problem 4: Bounds on Prefix Codes (20pt)

A. Prove the first part of the Kraft-McMillan inequality for Prefix Codes. In particular show that for any prefix code C,

$$\sum_{(s,w)\in C} 2^{-l(w)} \leq 1.$$

B. Prove that if you have $n = 2^k$ codewords in a prefix code and that if one of them is shorter than k bits, then at least two must be longer than k bits.

Problem 5: PPM (20pt)

In this exercise you will implement your own version of the *model-part* of PPM-C algorithm (see the required reading for compression). You may use any programming or scripting language of your choice. Attach your code to the writeup. The performance of your code does not matter, but be careful of the correctness. *Your code must implement the optimization described in the last paragraph on page 31 on "Introduction to data Compression".*

(a) Implement the PPM-C algorithm which takes the k value as a parameter. Your code does not need to implement an encoder, but it should output (1) a log of each message (including escape-characters) and its probability that it would pass to a coder (2) the total sum of bits of the encoded messages (use the theoretic non-rounded number of bits). Here is an example output:

```
b,0.15
a,0.2
$,0.4
u,0.18
..
y,0.45
Total bits:  93232.3
```

(b) Each student will have unique input: Navigate with your web browser to page `http://multi6.aladdin.cs.cmu.edu:3001/algoreal/` and enter your Andrew username and submit.

You will generated a roughly 140-word input, which you will use for the next question. The input is unique for each student. (Note: first word of the input is your username, do not remove it).

(c) Provide the theoretical number of bits encoded for your input using $k = 1, 2, 3, 4, 5$.

(d) For $k = 3$, provide the first and last 10 lines of your message log.

(e) Using $k = 3$, does the total number of bits change if you reverse the order of characters in the input? Explain (max. two sentences).