M525/P622 Quantum Information Ninnat Dangniam

Homework Assignment 1

50 points DUE: Thursday 4 Jul

1. Probability axioms (10 points).

In the lectures, we gave the four probability axioms:

1. $p(A) \geq 0$,
2. A is certain $\iff p(A) = 1$,
3. $p(A \vee B) = p(A) + p(B)$ if A and B are mutually exclusive events i,e, $A \wedge B = \emptyset$,
4. $p(B|A) = p(B \wedge A)/p(A)$.

From these axioms and Boolean arithmetic, derive the following probability laws.

(a) $p(\overline{A}) = 1 - p(A)$

(b) $p(A \vee B) = p(A) + p(B) - p(A \wedge B)$.

---

2. Monty Hall (10 points).

On the television show *Let's Make a Deal*, the host, Monty Hall, would show a contestant three doors. Behind one door was the grand prize, and behind the other two was nothing. The contestant chose one of the doors, say number 1, after which Monty would not open the chosen door yet. Instead, he opened one of the remaining doors, say number 3, revealing that there is nothing behind the door. The contestant was then offered the opportunity to switch to door number 2. Should he switch?

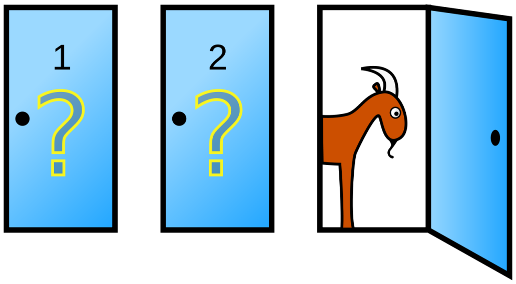

Figure 1: A popular rendition of the Monty Hall problem, with a car behind the correct door (no, you can't see it here) and a goat behind each remaining door. (Source: Wikipedia.)

Consider a generalization of the problem. There are N doors and only one door leads to the prize. The contestent guesses the door, Monty then opens n of the other doors, showing that none of them hides the prize, and the contestant is given the chance to switch to any of the $N - n - 1$ remaining doors.

(a) What is the probability $p(\checkmark)$ that the contestant's initial guess is right, and the probability $p(\text{✗})$ that the initial guess is wrong?

After Monty's revelation, Let G denotes the door of the contestant's initial **guess**, and let R denotes any one of the **remaining** doors.

(b) Given that the initial guess is right, what is the conditional probability $p(R|\checkmark)$ that the prize lies behind door R and the conditional probability $p(G|\checkmark)$, that the prize lies behind door G? What are the corresponding conditional probabilities, $p(R|\text{✗})$ and $p(G|\text{✗})$, given that the initial guess is wrong?

(c) Since the contestant doesn't know whether his initial guess is right or wrong, what is relevant to his decision are the unconditioned probabilities $p(R)$ and $p(G)$. What are $p(R)$ and $p(G)$? For what values of n should the contestant change his initial guess to one of the remaining doors?

---

3. Mutual information of three random variables? (10 points).

Given three random variables X, Y, and Z, one is tempted to define a "mutual information" for all three variables, $H(X:Y:Z) \equiv H(X:Y) - H(X:Y|Z) = H(X) - H(X|Y) - H(X|Z) + H(X|Y,Z)$, meant to be the amount of information shared by all three variables.

(a) Using a three-variable version of the Venn diagram in Nielsen and Chuang Fig. 11.2, justify the above definition.

(b) Show that this $H(X:Y:Z)$ can be negative (and thus is unsuitable as a measure of shared information) by considering the example of three binary variables with probabilities

$$p_{100} = p_{010} = p_{001} = \frac{1}{3},$$

and otherwise zero.

---

4. Prefix-free codes (20 points).

In this problem, we formulate a rigorous way of thinking of the Shannon entropy as the number of yes/no questions required to identify a particular alternative.

Let X be a random variable that can take on values x_j, $j = 1, \dots, D$, and let p_j be the probability of x_j. A sequence of yes(1)/no(1) questions can be depicted as a binary tree. A branch point is called a *node*, a path descending from a node is a *branch*, and a terminal node is a *leaf*, each leaf thus representing each alternative x_j. The sequence of answers leading to a leaf is a code word for the value at that leaf. The number of yes/no question leading to the code word x_j is the *length* l_j. In Figure 2 below, the code words and the corresponding lengths are $c_1 = 0$ with $l_1 = 1$, $c_2 = 100$ with $l_2 = 3$, $c_3 = 101$ with $l_3 = 3$, $c_4 = 11$ with $l_4 = 2$. The longest length in a code is the *order* L of the tree.

Such a code (collection of code words) is called a *prefix-free* code, because by construction no code word is a prefix of any other code word. Transmitting a message in a prefix-free code does not require an additional symbol to separate code words, like the space in English language. In particular, the prefix-free property guarantees that each code word in the message can be decoded as soon as it is received without having to wait to see if some other code word has the current code

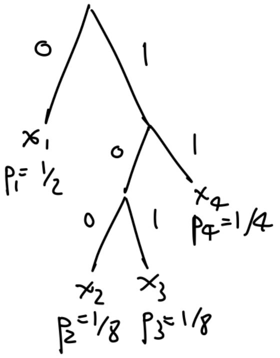

Figure 2: A tree of yes-no questions of order $L = 3$ for four alternatives.

word as a prefix. Thus, prefix-free codes are also called *instantaneous codes*.

(a) Let l_j be the length of the code word for x_j in some code, which *a priori* can be an arbitrary code. Show that if the *Kraft inequality* holds,

$$\sum_{j=1}^{D} 2^{-l_j} \leq 1, \tag{1}$$

then a prefix-free code with code-word length l_j, $j = 1, \ldots, D$ exists.

(b) Show that the average code-word length satisfies $\bar{l} \geq H(p)$.

(c) Show that it is possible to find code words such that $\bar{l} < H(p) + 1$.

The mismatch between the lower bound in **(b)** and the upper bound in **(c)** comes from the fact that probabilities p_j aren't generally inverse powers of 2. An *optimal code* is the one that has the smallest average length $\bar{l}$. There is a procedure for constructing an optimal code called *Huffman coding*. Codes other than Huffman codes can be optimal, but you will show below that they can never have shorter average length than a Huffman code.

Huffman coding works as follows. Take the two least likely alternatives, say x_1 and x_2, and make the last letter in their code words 0 and 1. Now glue these two alternatives into a single alternative whose probability is the sum $p_1 + p_2$. Then repeat the same process for this new set of alternatives. A particular case of Huffman coding is illustrated in Figure 3 below.

(d) Show that Huffman coding is optimal, i.e., that no code has an average code-word length smaller than that for Huffman coding. (**Hint**: One way to do this is to assume that we have an optimal code, and then convert it to a Huffman code.)

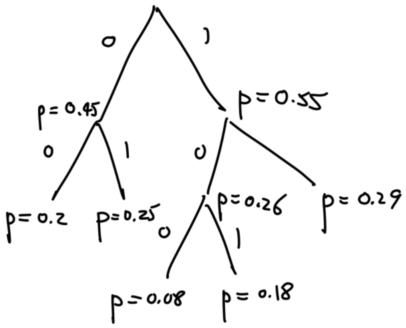

Figure 3: Huffman code for five alternatives with probabilities 0.29, 0.25, 0.2, 0.18, and 0.08. The average code-word length is 2.26.