Rutgers University
The State University of New Jersey
College of Engineering
Department of Electrical and Computer Engineering

330:542 Information Theory and Coding

Problem Set 3 Solutions

1. C&T problem 3.3

   (a) $P_0 = .995$ and $P_1 = .005$
   3 or fewer 1's out of 100

   $$\binom{100}{3} + \binom{100}{2} + \binom{100}{1} + \binom{100}{0} = 166,756 = N$$

   $\log_2 N = 17.34 \rightarrow$ need 18 bits.

   (b) Prob of Default (no codeword) $\rightarrow$ 1 - prob(3 or fewer 1's)

   $$= 1 - \sum_{i=1}^{n} \binom{100}{k} (0.005)^k (0.995)^{100-k} = 1.673 \times 10^{-3}$$

   (c) Chebyshev Prob$(\mid x - \overline{x} \mid > \epsilon) \leq \frac{\sigma^2}{\epsilon^2}$
   Average Number of 1's in source block:

   $$\begin{aligned} \overline{x} &= 100((0.005)(1) + (0.995)(0)) = 0.5 \text{1's /source block} \\ \sigma^2 &= 100((1-0.005)^2(0.005) + (0-0.005)^2(0.995)) \\ &= 100(0.00497) = 0.4975 \end{aligned}$$

   Prob$(\mid x - 0.5 \mid > 3.4\overline{999}$[1]$) \leq \frac{0.4975}{(3.4\overline{999})^2} = 0.0406$
   So bound works but is pretty loose.
   Extra : H(0.005)=0.045 bits
   so we need minimum of 4.5 bits to code 100 source letters

2. C&T 4.6

   $1 \rightarrow$ duration $1 \rightarrow$ prob P_1
   $2 \rightarrow$ duration $2 \rightarrow$ prob $P_2 = 1 - P_1$

[1] gives tightest bound

Source is memoryless and used to maximize:

$$\begin{aligned}
Q &= \frac{H(X)}{E(l_\alpha)} \\
E(l_\alpha) &= P_1 + 2P_2 = 1 + P_2 = 2 - P_1 \\
H(\mathrm{x}) &= -[P_1 log P_1 + (1-P_1) log(1-P_1)] \\
Q &= \frac{-P_1 log P_1 + (1-P_1) log(1-P_1)}{2-P_1}
\end{aligned}$$

$\frac{1}{2-P_1} \to$ monotone $\uparrow$ in P_1 max at $P_1 = 1$

H(X) $\to$ monotone $\uparrow$ in P_1, $P_1 < 0.5$ and

H(X) $\to$ monotone $\downarrow$ in P_1, $P_1 > 0.5$

so Q_{max} has $P_1 \geq 0.5$

$\frac{dQ}{dP_1} = \frac{-[1+log P_1 - (1+log(1-P_1))]}{2-P_1} + \frac{H(X)}{(2-P_1)^2}$ =0

Thus we get

(2-P_1)log($\frac{1-P_1}{P_1}$) + $P_1 log(\frac{1}{P_1})$ + $(1-P_1)$log($\frac{1}{1-P_1}$) $= 0$

log(1-P_1) +2log($\frac{1}{P_1}$) $= 0$

$\frac{1-P_1}{P_1^2} = 1$

$P_1^2 + P_1 - 1 = 0 \to P_1 = \frac{-1\pm\sqrt{5}}{2}$

Taking $P_1 = .618$ we get

$Q_{max} = \frac{.959}{2-.618} = .694$ bits/sec

3. C&T 4.14

p(X) is defined over 1,2,...,m. i.i.d sequence $X_0, X_1, \ldots$

$$N \equiv \min_n (X_n = X_0)$$

(a) given $X_0 = i$ prob(n=1)= $p_i \to$ prob(n=2)=(1-p_i)(p_i) $\to$ prob(n=k)=$(1-p_i)^{k-1}p_i$

so

$$prob(n=k) = \sum_{i=1}^{m} p_i^2 (1-p_i)^{k-1}$$

(add up all possibilities for X_0)

$$E(n) = \sum_{i=1}^{m}\sum_{n=1}^{\infty} np_i^2(1-p_i)^{n-1}$$
$$= \sum_{i=1}^{m} \frac{p_i^2}{(1-p_i)} \sum_{n=1}^{\infty} n(1-p_i)^n$$
$$= \sum_{i=1}^{m} \frac{p_i^2}{1-p_i}\frac{1-p_i}{p_i^2}$$
$$= \sum_{i=1}^{m} 1 = m$$

(b) Badly worded problem, Sorry!

$$E(log(n)) = \sum_{n=1}^{\infty} log(n) \sum_{i=1}^{m} p_i^2(1-p_i)^{n-1}$$

$$= \sum_{i=1}^{m} p_i \sum_{n=1}^{\infty} p_i(1-p_i)^{n-1} log(n)$$

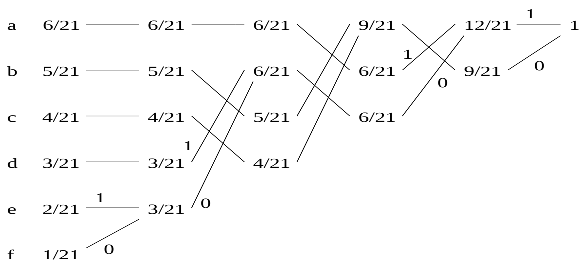
$p_i(1-p_i)^{n-1}$ is a geometric distribution. The second summation needs to be $\log(\frac{1}{p_i}) \rightarrow$ NOPE!
So this does not work! So what must be meant is

$$E(n/i) = \sum_{n=1}^{\infty}(n)p_i(1-p_i)^{n-1} = \frac{1}{p_i}$$

$$E_i(log(E(n|i)) = \sum_{i=1}^{\infty} p_i log(\frac{1}{p_i}) = H(X)$$

Problem was really,

$$E(log(n)) \leq H(X)$$
$$E(logn) = \sum_i p_i E(log(n)|i)$$
$$\leq \sum_i p_i log E(n|i) = \sum_i p_i log(\frac{1}{p_i}) = H(X)$$



a	11	6/21	12/21
b	01	5/21	10/21
c	00	4/21	8/21
d	101	3/21	9/21
e	1001	2/21	8/21
f	1000	1/21	4/21
			51/21 = E(L)

Figure 1: THIS IS FIG 5.14 (a)

4. C&T 5.11

   Suffix code is just a prefix code in reverse so everything follows as for preifx code.

   QED.

5. C&T 5.14

   (a) Please see the figure given above.

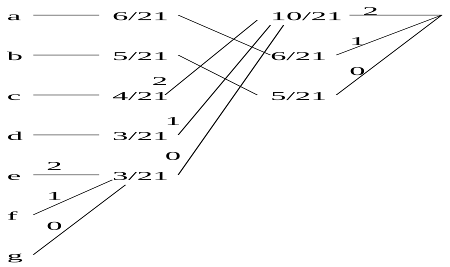
   (b) Ternary $\rightarrow$ 6=K(2) + 1 need to add 1 dummy. Please see the figure given below.

   (c) already done.

6. <u>Coding Problem</u>

   (a) Code 1 satisfies prefix, code 2 does not. Please see the tree graph shown below.

   (b) Both codes are uniquely decodable (try it) [Answers heuristic of part(d)] The 1 in code 2 serves as a spacer. The real info is contained in the number of consecutive zeros.



a	1	6/21	6/21
b	0	5/21	5/21
c	22	4/21	8/21
d	21	3/21	6/21
e	202	2/21	6/21
f	201	1/21	3/21
g	200	0	0
			34/21=E(L)

Figure 2: THIS IS FIG 5.14 (b)

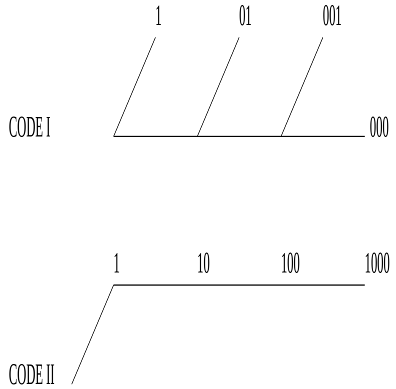


Figure 3: THIS IS FIGURE FOR CODING PROBLEM (a)

(c) $Code1 \rightarrow I(a_1:1) = log(\frac{p(a_1|1)}{p(a_1)}) = -log(p(a_1)$ since $p(a_1|1) = 1$

$Code2 \rightarrow I(a_1:1) = log(\frac{p(a_1|1)}{p(a_1)}) = 0$ since $p(a_1|1) = p(a_1)$

(d) Average:

$$\sum_{i=1}^{4} \sum_{j=0,1}^{4} log(\frac{p(a_i X_j)}{p(a_i)}$$

Code 1

$p(a_1 X_0) = 0$

$p(a_1 X_1) = 0.4$

$p(a_2 X_0) = 0.3$

$p(a_3 X_0) = 0.2$

$p(a_4 X_0) = 0.1$

$p(a_2 X_1) = 0$

$p(a_3 X_1) = 0$

$p(a_4 X_1) = 0$

$\overline{I} = p(a_1 X_1) log(\frac{p(a_1|X_1)}{p(a_1)})$

$+ p(a_2 X_0) log(\frac{p(a_2|X_0)}{p(a_2)})$

$+ p(a_3 X_0) log(\frac{p(a_3|X_0)}{p(a_3)})$

$+ p(a_4 X_0) log(\frac{p(a_4|X_0)}{p(a_4)})$

=-0.4log(0.4)-(0.3+0.2+0.1)log(0.6)

=-((0.4)log(0.4)+ (0.6)log(0.6))

see answer to (b) for heuristic

7. Kraft and Infinite codes

(a) First g(C)=1 $\rightarrow$ complete

Pf Say code is not complete then we can add a codeword to C to get C'

$g(C') = g(C) + 2^{-|C'|}$ where C' is the added codeword

C' is prefix free so $g(C') \leq 1$ but $g(C') = g(C) + 2^{-|C'|} \leq 1$ Thus, $g(C) < 1 \rightarrow$ not complete $\rightarrow g(C) < 1$

Therefore, g(C) = 1 $\rightarrow$ complete (since can always add a codeword otherwise)

Next Let code be finite i.e $|C| < \infty$

complete $\rightarrow$ g(C)=1 $\rightarrow$ let's prove $g(C) < 1 \rightarrow$ incomplete

Let m=max codeword length and let T_m be the complete binary tree with m levels .

2^m nodes at terminus of tree. Each codeword blocks off $2^{m-|C_i|}$ terminal nodes, so if

$g(C) < 1 \rightarrow$ we've got terminal nodes left. therefore $\rightarrow$ incomplete.

Therefore g(C)=1 $\rightarrow$ complete

(b) Proof of part(a) works for g(C)=1 $\rightarrow$ complete

Backward direction does not work because we can't make the 2^m argument anymore because m $\rightarrow \infty$

(c) This means that C can represent any binary string (covers all possible strings)

We want C full $\leftrightarrow$ C complete

We prove C not full $\rightarrow$ C incomplete

and C incomplete $\rightarrow$ C not full

i. not full, then there is a sequence not covered by C so we could add it to C and not violate prefix, that means C incomplete

ii. C not complete then there is a codeword which is not a prefix of any in C and has no prefix in C
Whoops!! $\rightarrow$ that means its uncovered
Therefore C not full
QED

(d) Construction: descend to level N, pick two nodes from either side of the tree and make the codewords

$C_1 = 0 \ldots 0$ (length N)

$C_2 = 1 \ldots 1$ (length N)

$\rightarrow$ these block off 2^{-N} off the tree each.

Then go to the next level and choose two more codewords

$C_3 = 01 \ldots 1$ (length $N+1$)

$C_4 = 10 \ldots 0$ (length $N+1$)

together block off 2^{-N} of tree.

Next layer $C_5 = 001 \ldots 1$ (length $N+2$)

$C_6 = 110 \ldots 0$ (length $N+2$)

together block off $2^{-(N+1)}$ of tree

Next

$C_7 = 010 \ldots 0$ (length N+3)

$C_8 = 101\ldots1$ (length N+3) together block off $2^{-(N+2)}$ of tree

You just have to make sure you don't choose a previous codeword as a prefix. But this is easy because you only choose two of the possible 2^k codewords of length k at any given step.

So eventually all possible sequences appear as prefixes to codewords or as codewords. So the code is full. i.e codewords contain all possible sequences and we can't add a codeword without it being a prefix of one already chosen. Therefore our code is complete.

However, to calculate $g(C)$ we consider the portion of tree blocked is

$$g(C) = \sum_{i=0}^{\infty} 2^{-(N-1+i)} = 2^{(N-1)} \frac{1}{1-2^{-1}} = 2^{-N} < 1$$

and we can choose N to make $g(C)$ as small as desired. So completeness does not imply $g(C) = 1$ for infinite codes.