Exercises 1 (solutions) Jan. 2008

1. No. We showed in class that, for any prefix code, $\sum_{i=1}^{N} 2^{-\lambda_i} \leq 1$. But for the numbers given in the question $\sum_{i=1}^{6} 2^{-\lambda_i} = \frac{33}{32} > 1$. Therefore, no such prefix code exists.

2. (a) From the definition of entropy :

$$H == 0.625 \log \frac{1}{0.625} + 0.125 \log \frac{1}{0.125} + 0.25 \log \frac{1}{0.25} \approx 1.3$$

   (b) Here are the four possible codes that can be constructed

   - $C(A_1) = 1, C(A_2) = 00, C(A_3) = 01$
   - $C(A_1) = 1, C(A_2) = 01, C(A_3) = 00$
   - $C(A_1) = 0, C(A_2) = 10, C(A_3) = 11$
   - $C(A_1) = 0, C(A_2) = 11, C(A_3) = 10$

   In all cases we obtain the same code length. From the definition of the average code length

$$\overline{\lambda} = \sum_{i=1}^{3} \lambda_i p(A_i) = 1 * 0.625 + 2 * 0.125 + 2 * 0.25 = 1.375 \text{bits/Symbol}$$

   (c) $p(A_1, A_2) = .625 * .125,\ p(A_1 A_3) = .625 * .25,\ p(A_2 A_3) = .125 * .25,\ etc$

   Here is one Huffman code:

   $C(A_1A_1) = 1$
   $C(A_1A_2) = 0110$
   $C(A_1A_3) = 000$
   $C(A_2A_1) = 0111$
   $C(A_2A_2) = 010110$
   $C(A_2A_3) = 010111$
   $C(A_3A_1) = 001$
   $C(A_3A_2) = 01010$
   $C(A_3A_3) = 0100$

   The code length per symbol is $\overline{\lambda}/2$.

$$\overline{\lambda}/2 = \sum_{i=1}^{3} \sum_{j=1}^{3} \lambda_{i,j} \;\; p(A_{i,j}) \approx 2.64/2 = 1.32 \quad \text{bits/Symbol}$$

3. Yes. Following the argument in class, we can define a prefix code such that $\lambda_i = \lceil \log \frac{1}{p(A_i)} \rceil$. The Huffman code has average code length less than or equal to this prefix code (since Huffman is optimal). Thus,

$$\begin{aligned} \overline{\lambda}_{Huff} &\leq \sum p(A_i)\lambda_i \\ &= \sum_I p(A_i)\lceil \log \frac{1}{p(A_i)} \rceil \\ &< \sum_i p(A_i)(\log \frac{1}{p(A_i)} + 1) \\ &= H + 1 \end{aligned}$$

4. (a) $C(A_4) = 00, C(A_3) = 01, C(A_1) = 10, C(A_6) = 1100, C(A_2) = 1101, C(A_5) = 111$

   (b) $A_3 A_1 A_4 A_5 A_2 A_6$

5. (a) For the average code length to equal the entropy, it is sufficient that $\lambda(A_i) = \log \frac{1}{p(A_i)}$ for all i, and so $p(A_i) = 2^{-\lambda_i}$. Thus,

$$p(A_1) = \frac{1}{2}, \quad p(A_2) = \frac{1}{8}, \quad p(A_3) = \frac{1}{8}, \quad p(A_4) = \frac{1}{4}$$

   (b) For average code length to be different from the entropy, but still be a Huffman code, we could just change the probabilities slightly. The easiest way to do this is to perturb the probabilities of the two least probable symbols, so that they remain the two least probable symbols (and do not change the Huffman code). For example,

$$p(A_1) = \frac{1}{2}, \quad p(A_2) = \frac{1}{8} + \frac{1}{100}, \quad p(A_3) = \frac{1}{8} - \frac{1}{100}, \quad p(A_4) = \frac{1}{4}$$

   Why should this work? The key idea is to examine how the pertubation affects the entropy. If we just look at the contribution to the entropy from the two least probable terms (of probability p_0), we get:

$$\begin{aligned}
& (p_0 - \epsilon)\log\frac{1}{p_0 - \epsilon} + (p_0 + \epsilon)\log\frac{1}{p_0 + \epsilon} \\
= \;& 2p_0\left(\frac{p_0 - \epsilon}{2p_0}\log\frac{1}{p_0 - \epsilon} + \frac{p_0 + \epsilon}{2p_0}\log\frac{1}{p_0 + \epsilon}\right) \\
< \;& 2p_0 \log\left(\frac{1}{2p_0} + \frac{1}{2p_0}\right) \\
= \;& 2p_0 \log\frac{1}{p_0}
\end{aligned}$$

   Here we have used the strict inequality for Jensen's. This requires slightly modifying the statement and proof of Jensen's inequality seen in class, so that it applies to strictly convex functions (such as log).

   (c) For a Huffman code, A_2 and A_3 would have to be the symbols with lowest probabilities. So choose any set of probabilities, which makes this condition fail. For example, $p(A_2) = 0.7, \quad p(A_1) = 0.1, \quad p(A_3) = 0.1, \quad p(A_4) = 0.1$

6. (a) The codeword lengths of all the symbols can differ by at most one bit. The proof is by contradiction. Suppose that two codeword lengths were λ_0 and $\lambda_0 + l$, where $l \geq 2$. Replace the (leaf) node at height λ_0 with an internal node and give it two children, namely the former leaf nodes at heights λ_0 and $\lambda_0 + l$, respectively. Thus, we have replaced two leaf nodes at height λ_0 and $\lambda_0 + l$ with two leaf nodes at heights $\lambda_0 + 1$. Since, $l \geq 2$ and since all symbols have equal probability, we have certainly not increased the average code length.

   Now we note that the node in the original tree that was the sibling of the node at height $\lambda_0 + l$ no longer has a sibling. Thus, we can shorten the branch leading to this node, thereby strictly decreasing the average code length. Hence the original code could not have been optimal prefix code.

(b) Again, the codeword lengths of these M symbols can differ by at most one bit. The same argument applies as above.

We cannot say anything more. Even if these M symbols have probability that is a power of 2, this does not guarantee that all symbols have the same codeword length.

e.g. $N = 4$ and $M = 3$. $p(A_1) = \frac{13}{16}$, $p(A_i) = \frac{1}{16}$ for $i = 2, 3, 4$

7. (a) From the definition of entropy :

$$H = \sum_{i=1}^{6} p(A_i) \log \frac{1}{p(A_i)} = \sum_{i=1}^{6} \frac{1}{6} \log 6 = \log 6 \simeq 2.58$$

(b) Considering that events X_1 and X_2 are independent, then $p(X_1, X_2) = p(X_1) * p(X_2)$. Hence $p(X_1, X_2) = \frac{1}{6}\frac{1}{6} = \frac{1}{36}$. Once again from the definition of entropy:

$$H = \sum_{i=1}^{36} p(A_i) \log \frac{1}{p(A_i)} = \sum_{i=1}^{36} \frac{1}{36} \log 36 = \log 36 \simeq 5.17$$

Note that the answer to (b) is twice that of (a).

(c) We want to find the entropy of Y which is equal to the sum of X_1 and X_2. First notice that since Y is the sum of two dice, it can take a value between 2 and 12. The probability for each of these values is :

$p(Y = 2) = p(X_1 = 1) * p(X_2 = 1) = \frac{1}{36}$

$p(Y = 3) = [p(X_1 = 1) * p(X_2 = 2)] + [p(X_1 = 2) * p(X_2 = 1] = \frac{2}{36}$

Similarly:

$p(Y = 4) = \frac{3}{36}$

$p(Y = 5) = \frac{4}{36}$

$p(Y = 6) = \frac{5}{36}$

$p(Y = 7) = \frac{6}{36}$

$p(Y = 8) = \frac{5}{36}$

$p(Y = 9) = \frac{4}{36}$

$p(Y = 10) = \frac{3}{36}$

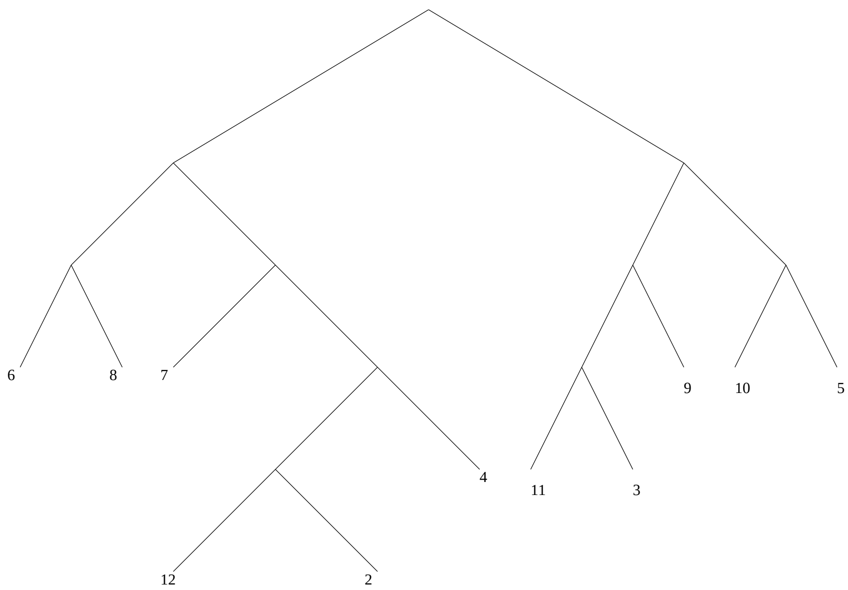
$p(Y = 11) = \frac{2}{36}$

$p(Y = 12) = \frac{1}{36}$

$$H = 2(\frac{1}{36} \log 36 + \frac{2}{36} \log \frac{36}{2} + \frac{3}{36} \log \frac{36}{3} + \frac{4}{36} \log \frac{36}{4} + \frac{5}{36} \log \frac{36}{5}) + \frac{6}{36} \log \frac{36}{6} \simeq 3.27$$

(d) Constructing the Huffman code, we follow the Huffman algorithm provided in the course notes. Note that you may obtain a different Huffman tree, using the algorithm provided in class, since due to equal probabilities, you may have a choice in the order of the merger of the symbols. However you must obtain the same average code length regardless of the order of mergers. Here is an example of the Huffman tree that you may obtain. The right branches are 1's and the left branches are 0's.

The average code length for the above Huffman code is therefore: $\sum_{i=2}^{12} p(A_i)\lambda_i = 3.306$, where we have the $p(A_i)$ from part C, and the λ_{A_i} are taken from the tree.



8. Average code length of the unary code is

$$\frac{1}{256}\sum_{i=1}^{256} i = (256 * 257/2)/256 = 257/2.$$

Average code length of the Elias1 code is

$$\frac{1}{256}\sum_{i=1}^{256} 2\lfloor \log(i) \rfloor + 1 \leq \frac{1}{256}\sum_{i=1}^{256} 2\lfloor \log(256) \rfloor + 1 = 17.$$

Thus the average code length of the Elias1 code is clearly shorter in this case.

9. (a)

$$C(A_1) = 01, \quad C(A_2) = 0000, \quad C(A_3) = 10, \quad C(A_4) = 001, \quad C(A_5) = 0001, \quad C(A_6) = 11$$

(this is just one example, the important thing is the codeword length)

(b) We automatically get an optimal prefix code if $p(A_i) = 2^{-\lambda_i}$, since in this case the average code length is equal to the entropy. This gives us the following probabilities:

$$p(A_1) = 2^{-2}, \quad p(A_2) = 2^{-4}, \quad p(A_3) = 2^{-2}, \quad p(A_4) = 2^{-3}, \quad p(A_5) = 2^{-4}, \quad p(A_6) = 2^{-2}$$

And an entropy of:

$$\begin{aligned} H &= \sum_{i=1}^{N=6} p(A_i) \log \frac{1}{p(A_i)} \\ &= 2^{-2} \log \frac{1}{2^{-2}} + 2^{-4} \log \frac{1}{2^{-4}} + 2^{-2} \log \frac{1}{2^{-2}} + 2^{-3} \log \frac{1}{2^{-3}} + 2^{-4} \log \frac{1}{2^{-4}} + 2^{-2} \log \frac{1}{2^{-2}} \\ &= etc \end{aligned}$$

(c) We perturb the probabilities of the two least probably symbols by a small value (ϵ), maintaining the property $\sum_{i=1}^{N} p(i) = 1$ and that the two least probable symbols remain the two least probable symbols.

$$p(A_1) = 2^{-2}, \;\; p(A_2) = 2^{-4}+\epsilon, \;\; p(A_3) = 2^{-2}, \;\; p(A_4) = 2^{-3}, \;\; p(A_5) = 2^{-4}-\epsilon, \;\; p(A_6) = 2^{-2}$$

where ϵ is some small number, say $\epsilon = \frac{1}{100}$, for example.
It remains for you to show that the average code length is strictly greater the entropy. For this, you can use a strict inequality version of Jensen's inequality to compare $\log \frac{1}{2^{-4}+\epsilon} + \log \frac{1}{2^{-4}+\epsilon}$ with $\log \frac{1}{2^{-4}} + \log \frac{1}{2^{-4}}$. Details left to you.)

10. *Advantage*: Any choice of b implies a specific probability distribution on the occurences of integer i's. For any given source of data, we wish to choose b such that the code is near optimal for that source (i.e. the average code length is close to the entropy). If we restrict b to be a power of 2, then we are limiting ourselves in being able to fit the probabilities for our source. (It could be that the best-fitting b is indeed a power of 2, but it could also be that the best-fitting b is some integer that is not a power of 2.) Thus, by allowing b to be some integer that is not a power of 2, we may be able to better fit the probabilities of the data.

    Disadvantage: If b is not a power of 2, then each time we specify "which element within a group", we have a codeword which never gets used. e.g. If b=3 (groups of 3), then we need 2 bits and we use 00,01,10 to say which element in the group, but don't ever use 11. This would seem to be a source of inefficiency.

11.

$$\begin{aligned}
\log(N!) &= \log 1 + \log 2 + ... + \log N \\
&= \sum_{i=1}^{N} \log i \\
&= N \sum_{i=1}^{N} \frac{1}{N} \log i \\
&\leq N \log \big(\sum_{i=1}^{N} \frac{i}{N}\big) \\
&= N \log \big(\frac{N(N+1)}{2N}\big) \\
&= N \log \big(\frac{N+1}{2}\big)
\end{aligned}$$

$$\begin{aligned}
\sum_{i=N_1}^{N_2} \log i &= \sum_{i=N_1}^{N_2} \log i \\
&= (N_2 - N_1 + 1) \sum_{i=N_1}^{N_2} \frac{1}{(N_2 - N_1 + 1)} \log i \\
&\leq (N_2 - N_1 + 1) \log \left(\sum_{i=N_1}^{N_2} \frac{i}{(N_2 - N_1 + 1)}\right) \\
&= (N_2 - N_1 + 1) \log \left(\frac{1}{(N_2 - N_1 + 1)} \sum_{i=N_1}^{N_2} i\right) \\
&= (N_2 - N_1 + 1) \log \left(\frac{1}{(N_2 - N_1 + 1)} \left(\frac{(N_2 + N_1)(N_2 - N_1 + 1)}{2}\right)\right) \quad \text{(check for yourself)} \\
&= (N_2 - N_1 + 1) \log \left(\frac{(N_2 + N_1)}{2}\right)
\end{aligned}$$

12. (a)

$$\sum_{j=1}^{g} = \frac{g(g+1)}{2} = i$$

$$g^2 + g - 2i = 0$$

$$g = \frac{-1 \pm \sqrt{1+8i}}{2} \approx \sqrt{2i} - \frac{1}{2}$$

Now, using a unary code for g:

$$\begin{aligned}
\lambda_i &\approx \sqrt{i} + \lceil \log g \rceil \\
&\approx \sqrt{i} + \log \sqrt{2i} \\
&= \sqrt{i} + \frac{1}{2}(1 + \log i)
\end{aligned}$$

(b) The probabilities $p(i)$ that would make the average code length roughly equal to the entropy are $p(i) \approx 2^{-\lambda_i}$.

$$\begin{aligned}
2^{-\lambda_i} &\approx 2^{-(\sqrt{i} + \frac{1}{2}\log 2i)} \\
&= 2^{-\sqrt{i} - \frac{1}{2}\log 2i} \\
&= (2^{-\sqrt{i}})(2^{-\frac{1}{2}\log 2i}) \\
&= 2^{-\sqrt{i}}\left(\frac{1}{\sqrt{2i}}\right) \quad \square
\end{aligned}$$

(c) Using a Golomb code for g with parameter b:

$$\begin{aligned}
\lambda_i &\approx \lceil\frac{\sqrt{i}}{b}\rceil + \log b + \frac{1}{2}\log 2i \\
2^{-\lambda_i} &\approx 2^{-(\frac{\sqrt{i}}{b}+\log b+\frac{1}{2}\log 2i)} \\
&= 2^{-\frac{\sqrt{i}}{b}-\log b-\frac{1}{2}\log 2i} \\
&= (2^{-\frac{\sqrt{i}}{b}})(2^{-\log b})(2^{-\frac{1}{2}\log 2i}) \\
&= (2^{-\frac{\sqrt{i}}{b}})(\frac{1}{b})(\frac{1}{\sqrt{2i}}) \quad \square
\end{aligned}$$