Concept Inventory:

- Measuring information content; entropy
- Two's complement; modular arithmetic
- Variable-length encodings; Huffman's algorithm
- Hamming distance, error detection, error correction

Notes:

Measuring information: \( I(x_i) = \log_2(1/p_i) \) bits

\( N \) equally-probable choices down to \( M \) choices: \( \log_2(N/M) \) bits

Entropy: \( H(X) = E(I(X)) = \sum_i p_i \log_2(1/p_i) \)

N-bit 2's complement:

\[ \begin{array}{cccccccc}
\cdots & \cdots & 2^3 & 2^2 & 2^1 & 2^0 \\
\end{array} \]

"sign bit"

Range: \( -2^{N-1} \) to \( 2^{N-1} - 1 \)

Variable-length encoding:
Symbols with smallest \( p_i \) have longest encodings, symbols with largest \( p_i \) have shortest encodings.

Huffman's algorithm:
- Build binary decoding tree bottom-up starting with symbols that have smallest \( p_i \),
- Each step: combine the two symbols or subtrees with smallest \( p_i \), into new subtree.

Hamming distance:
- \( HD = \# \) of bit positions that differ between to codewords
- need to know min Hamming distance (\( HD_{\text{min}} \)) considering all pairs of codewords
- \# of errors detected = \( HD_{\text{min}} - 1 \)
- \# of errors corrected = \( \left\lfloor \frac{HD_{\text{min}}-1}{2} \right\rfloor \)
1. Information Content and Entropy

A. You are given an unknown 3-bit binary number. You are then told that the binary representation contains exactly two 1's. How much information have you been given?

- 3 bits ⇒ 8 choices
- Exactly two 1's ⇒ 011, 101, 110 ⇒ 3 choices

\[ \log_2(\frac{8}{3}) \text{ bits} \]

B. You are then given the additional information that the number is also odd. How much additional information have you been given?

- Odd ⇒ 011, 101 ⇒ 2 choices

\[ \log_2(\frac{3}{2}) \text{ bits} \]

C. A random variable X represents the outcome of flipping an unfair coin, where \( p(\text{HEADS}) = 0.6 \). Please give the value for the entropy \( H(X) \). You may express your answer as a numeric expression (i.e., you don’t have to actually do the arithmetic).

\[ H(X) = (0.6) \log_2 \left( \frac{1}{0.6} \right) + (0.4) \log_2 \left( \frac{1}{0.4} \right) = 0.97 \text{ bits} \]

D. A single decimal digit is chosen at random and you’re told that its value is 0 mod 3. How much information have you learned about the digit?

- Decimal digit ⇒ 10 choices
- 0 mod 3 ⇒ 0, 3, 6, 9 ⇒ 4 choices

\[ \log_2 \left( \frac{10}{4} \right) \text{ bits} \]

E. X is an unknown 8-bit binary number. You are given another 8-bit binary number, 10101100, and told that the Hamming distance between X and 10101100 is one. How many bits of information about X have you been given? You can give a formula if you wish.

- 8 bits ⇒ 256 choices
- 10 = 1 from 10101100 ⇒ 8 choices

\[ \log_2 \left( \frac{256}{8} \right) = 5 \text{ bits} \]

F. We wish to transmit messages comprised of the four symbols shown below with their associated probabilities and 5-bit fixed-length encoding.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>( p(\text{symbol}) )</th>
<th>encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>0.5</td>
<td>00000</td>
</tr>
<tr>
<td>( \beta )</td>
<td>0.125</td>
<td>11100</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>0.25</td>
<td>11011</td>
</tr>
<tr>
<td>( \delta )</td>
<td>0.125</td>
<td>10111</td>
</tr>
</tbody>
</table>

An unknown symbol is received and you are told it’s not \( \delta \). How much information have you received?

\[ p(\text{not } \delta) = \frac{3}{8} \Rightarrow \log_2 \left( \frac{1}{\frac{3}{8}} \right) = \log_2 \left( \frac{8}{3} \right) \text{ bits} \]
G. When transmitting a message comprised of these four symbols with the probabilities as given above, what is the expected amount information received when you are told the next symbol in the message?

\[
\text{Entropy} = (0.5) \log_2 \left( \frac{1}{5} \right) + (0.25) \log_2 \left( \frac{1}{25} \right) + 2(0.125) \log_2 \left( \frac{1}{125} \right)
\]

\[= 1.75 \text{ bits}.\]

H. You are given an unknown 5-bit binary number. You are then told that the first and last bits are the same. How much information have you been given?

- 5 bits \( \Rightarrow 32 \) choices
- 1st and last bits the same \( \Rightarrow 16 \) choices

\[\log_2 (32) = 5 \text{ bits}, \quad \log_2 (16) = 4 \text{ bits}.\]

I. I’ve randomly selected a letter from the alphabet and tell you that my selection is neither “X”, “Y”, nor “Z”. How much information have I given you about my letter?

- random letter \( \Rightarrow 26 \) choices
- not X, Y, Z \( \Rightarrow 23 \) choices

\[\log_2 (26) \approx 5 \text{ bits}, \quad \log_2 (23) \approx 5 \text{ bits}.\]

J. I make up a random 4-bit two’s complement number by flipping a fair coin to determine each bit. You’re trying to guess the number. If I tell you that the number is positive (\( > 0 \)), how many bits of information have I given you? Be precise; you may answer by a formula or a number.

- 4 bits \( \Rightarrow 16 \) choices
- positive \( \Rightarrow 7 \) choices

\[\log_2 (16) = 4 \text{ bits}, \quad \log_2 (7) \approx 3 \text{ bits}.\]
2. Two's Complement

A. What is the 6-bit two’s complement representation of the decimal number -21?

\[ \begin{align*}
21_{10} &= \quad 010101 \\
-21_{10} &= \bar{21}_{10} + 1 = 10101 + 1 = \boxed{10111}
\end{align*} \]

B. What is the hexadecimal representation for decimal -51 encoded as an 8-bit two’s complement number?

\[ \begin{align*}
51 &= \quad 00110011 \\
\bar{51} &= \quad 11001100 \\
\bar{51} + 1 &= (11001100 + 1) = 11001101 = \boxed{0xCD}
\end{align*} \]

C. The hexadecimal representation for an 8-bit two’s complement number is 0xD6. What is its decimal representation?

\[ \begin{align*}
0xD6 &= 11011010 = -128 + (64 + 4 + 2) = -42
\end{align*} \]

D. Since the start of official pitching statistics in 1988, the highest number of pitches in a single game has been 172. Assuming that remains the upper bound on pitch count, how many bits would we need to record the pitch count for each game as a two’s complement binary number?

\[ \text{Need 9 bits: } \quad 2^n = 2^8 = 256 > 172 \\
\]

E. Can the value of the sum of two 2’s complement numbers 0xB3 + 0x47 be represented using an 8-bit 2’s complement representation? If so, what is the sum in hex? If not, write NO.

\[ \begin{align*}
0xB3 &= 10110011 \\
+0x47 &= 01000111 \\
\hline
0xFA &= 11111010
\end{align*} \]

F. Can the value of the sum of two 2’s complement numbers 0xB3 + 0xB1 be represented using an 8-bit 2’s complement representation? If so, what is the sum in hex? If not, write NO.

\[ \begin{align*}
0xB3 &= 10110011 \\
+0xB1 &= 10110001 \\
\hline
10110 0100 & \text{ (Overflow!)}
\end{align*} \]
G. Please compute the value of the expression 0xBB – 8 using 8-bit two’s complement arithmetic and give the result in decimal (base 10).

\[
-8 = \overline{N(\overline{8})} = \overline{N(1011_2)} = \overline{1110_2} = 11110000 = -128 + 32 + 16 + 2 + 1 = -77
\]

H. What is the smallest (most negative) integer that can be represented as an 8-bit two’s complement integer? Give your answer as a decimal integer.

\[
\text{most negative} = 10000000 = -128
\]

I. The following operations are performed on an 8-bit adder. Give the 8-bit sum produced for each, in hexadecimal.

\[
\begin{array}{c}
0x\text{F}0 + 0x\text{3}4 = 0x2\text{4} \\
0x\text{F}0 + 0x\text{8}0 = 0x7\text{0}
\end{array}
\]

J. Using a 5-bit two’s complement representation, what is the range of integers that can be represented with a single 5-bit quantity?

\[
\text{range} = -2^4 \text{ to } 2^4 - 1 = -16 \text{ to } 15
\]

K. Consider the following subtraction problem where the operands are 5-bit two’s complement numbers. Compute the result and give the answer as a decimal (base 10) number.

\[
\begin{array}{c}
10101 \\
- 00011
\end{array}
\]

\[
10010 = -16 + 2 = -14
\]
3. Variable-length Encodings

A. Given a variable $X$ that can take on one of four values A, B, C, or D with the following probabilities.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.5</td>
</tr>
<tr>
<td>B</td>
<td>0.3</td>
</tr>
<tr>
<td>C</td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td>0.1</td>
</tr>
</tbody>
</table>

If you encoded this variable using a Huffman encoding, how many bits would be in the encoding of each of the symbols?

For each of the probability distributions for symbols A-E, select the Huffman encoding tree or trees that could result from running Huffman’s algorithm on those probability distributions.

B. $p(A) = 0.3$, $p(B) = 0.3$, $p(C) = 0.2$, $p(D) = 0.1$, $p(E) = 0.1$. Tree(s): 2

C. $p(A) = 0.6$, $p(B) = 0.1$, $p(C) = 0.1$, $p(D) = 0.1$, $p(E) = 0.1$. Tree(s): 3

D. $p(A) = 0.5$, $p(B) = 0.15$, $p(C) = 0.15$, $p(D) = 0.1$, $p(E) = 0.1$. Tree(s): 3

E. $p(A) = 0.5$, $p(B) = 0.2$, $p(C) = 0.15$, $p(D) = 0.05$, $p(E) = 0.1$. Tree(s): 1
Baseball loves statistics! There are many different types of pitches that a pitcher can throw – the table below shows the probability for each type of pitch during 2014.

<table>
<thead>
<tr>
<th>Type of pitch</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fastball</td>
<td>0.34</td>
</tr>
<tr>
<td>Change-up</td>
<td>0.14</td>
</tr>
<tr>
<td>Curveball</td>
<td>0.08</td>
</tr>
<tr>
<td>Slider</td>
<td>0.28</td>
</tr>
<tr>
<td>Other</td>
<td>0.16</td>
</tr>
</tbody>
</table>

F. How much information have you received when learning that particular pitch was NOT a fastball? You may express your answer as a formula if you wish.

\[
p(\text{not fastball}) = 1 - 0.34 = 0.66
\]

G. To save on storage costs, Major League Baseball would like to use an optimal variable-length code to record, one at a time, the type of each pitch (i.e., to record one of the 5 types shown in the table above). Use Huffman’s algorithm to derive such a code and list the resulting binary encodings below.

H. The table below shows the 2012-13 enrollments in the various EECS majors. To save a bit of space in their database the department would like to use a variable-length Huffman code to encode a student’s choice of major. For each of the four majors, please give the encoding the department should use.

<table>
<thead>
<tr>
<th>Major</th>
<th>Count</th>
<th>( p )</th>
<th>( p \log_2(1/p) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-1</td>
<td>74</td>
<td>0.09</td>
<td>0.30</td>
</tr>
<tr>
<td>6-2</td>
<td>387</td>
<td>0.44</td>
<td>0.52</td>
</tr>
<tr>
<td>6-3</td>
<td>360</td>
<td>0.41</td>
<td>0.53</td>
</tr>
<tr>
<td>6-7</td>
<td>54</td>
<td>0.06</td>
<td>0.25</td>
</tr>
<tr>
<td>Total</td>
<td>875</td>
<td>1.00</td>
<td>1.60</td>
</tr>
</tbody>
</table>

I. We wish to transmit messages comprised of the four symbols shown below with their associated probabilities and 5-bit fixed-length encoding

<table>
<thead>
<tr>
<th>Symbol</th>
<th>( p(\text{symbol}) )</th>
<th>encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>0.5</td>
<td>00000</td>
</tr>
<tr>
<td>( \beta )</td>
<td>0.125</td>
<td>11100</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>0.25</td>
<td>11011</td>
</tr>
<tr>
<td>( \delta )</td>
<td>0.125</td>
<td>10111</td>
</tr>
</tbody>
</table>

Huffman’s algorithm is used to construct a variable-length code for the four symbols for transmitting a single symbol at a time. The resulting encoding could be

1. \( \alpha:00, \beta:01, \gamma:10, \delta:10 \)
2. \( \alpha:00, \beta:01, \gamma:100, \delta:101 \)
3. \( \alpha:1, \beta:01, \gamma:000, \delta:001 \)
4. \( \alpha:0, \beta:110, \gamma:01, \delta:111 \)
5. none of the above
NerdLink is a new web-based startup that aims to keep MIT EECS students in touch with their parents. NerdLink streamlines parental communication by providing each student with an online choice of one of the five messages, then automatically fills in boilerplate and emails the parent a long and charming version of the message. The five messages, and their relative probabilities, are listed below:

<table>
<thead>
<tr>
<th>Message #</th>
<th>Message to parents</th>
<th>p(Message)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>Send money!</td>
<td>60%</td>
</tr>
<tr>
<td>M2</td>
<td>I love this course called 6.004</td>
<td>8%</td>
</tr>
<tr>
<td>M3</td>
<td>I’m changing my major to Poetry</td>
<td>2%</td>
</tr>
<tr>
<td>M4</td>
<td>I’m getting a 5.0 this term!</td>
<td>1%</td>
</tr>
<tr>
<td>M5</td>
<td>Nothing much is new… (none of the above)</td>
<td>29%</td>
</tr>
</tbody>
</table>

NerdLink’s initial implementation conveyed each message using a fixed-length code.

J. What is the average number of bits needed to convey a message, using a fixed-length code?

5 choices \(\Rightarrow\) need \(\lceil 3 \text{ bits} \rceil\) to encode

K. Given the probability distribution of the messages, what is the actual amount of information conveyed by message M5? Your answer may be a formula.

\[ p(M5) = 0.29 \quad H(M5) = \log_2 \left( \frac{1}{0.29} \right) \]

L. To enable error correction, the fixed-length code for a given message is sent five times. Using the five copies of the received message, in the worst case how many bit errors can be corrected at the receiver?

\[ \text{min } H_0 \text{ of original fixed-length code } = 3 \text{ bits} \quad \text{correction} = \left[ \frac{N_0 - 1}{2} \right] \]

\[ \text{min } H_0 \text{ of replication } 5 \text{ times } = 5 \text{ bits} \]

NerdLink, wanting to economize on communication costs, has hired you as a consultant to design a Huffman code for sending the messages.

M. Give the number of bits sent by your Huffman code for each message (M1 though M5), and the average number of bits transmitted per message using your code (a formula will be fine).
The Registrar's office would like to encode the letter grades (A, B, C, D, F) from a large GIR with 1000 students. They plan to encode each grade separately using a variable-length code. An analysis of previous terms has produced the following table of grade probabilities. In case it's useful, a thoughtful former 6.004 student has augmented the table by computing $p \log_2(1/p)$ for each grade.

<table>
<thead>
<tr>
<th>Grade</th>
<th>$p$</th>
<th>$p \log_2(1/p)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.24</td>
<td>0.49</td>
</tr>
<tr>
<td>B</td>
<td>0.35</td>
<td>0.53</td>
</tr>
<tr>
<td>C</td>
<td>0.21</td>
<td>0.47</td>
</tr>
<tr>
<td>D</td>
<td>0.13</td>
<td>0.38</td>
</tr>
<tr>
<td>F</td>
<td>0.07</td>
<td>0.27</td>
</tr>
<tr>
<td><strong>Totals</strong></td>
<td>1.00</td>
<td>2.14</td>
</tr>
</tbody>
</table>

N. Use Huffman's algorithm to construct an optimal variable-length encoding.

O. Two 6.004 students have proposed competing variable-length codes. Alice says that encoding 1000 grades using her code will, on the average, produce messages of 2200 bits. Bob says that encoding 1000 grades using his code will, on the average, produce messages of 1950 bits. Which of the following is your best response when the Registrar asks your opinion?

(A) Choose Bob's: it has the shorter average length
(B) Choose Alice's: more bits means more information is transmitted
(C) Choose Bob's: Bob's average message length is less than the information entropy
(D) Choose Alice's: Bob's average message length is less than the information entropy
(E) Choose neither: a fixed-length code will have lower average message size

Best Choice (A through E): **D**

Bobs code is ambiguous since it sends fewer bits than entropy lower bound
4. Error Detection and Correction

A. A message about the suit of a card is sent using the encoding shown at the right. Using this encoding, how many bit errors can be detected? How many bit errors can be corrected?

\[
\begin{align*}
\min HD &= 2 \\
\text{detected} &= HD - 1 = 1 \\
\text{corrected} &= \left\lfloor \frac{HD - 1}{2} \right\rfloor = 0
\end{align*}
\]

B. A message about the suit of a card is sent using the encoding shown at the right. Give an example of a 5-bit received message with an uncorrectable single-bit error or write NONE if all single-bit errors can be corrected.

\* Find 2 codewords with HD = 2: 
\* Introduce 1-bit error

\[
\text{Heart: 00000} \\
\text{Diamond: 11001} \\
\text{Spade: 10111} \\
\text{Club: 01011}
\]

C. The MIT baseball coach likes to record the umpire's call for each pitch (one of "strike", "ball" or "other"). Worried that bit errors might corrupt the records archive, he proposes using the following 5-bit binary encoding for each of the three possible entries:

<table>
<thead>
<tr>
<th>Strike</th>
<th>11111</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ball</td>
<td>01101</td>
</tr>
<tr>
<td>Other</td>
<td>00000</td>
</tr>
</tbody>
</table>

Using this encoding what is the largest number of bit errors that be detected when examining a particular record? The largest number of bit errors that can be corrected?

\[
\begin{align*}
\text{detected} &= HD - 1 = 1 \\
\text{corrected} &= \left\lfloor \frac{HD - 1}{2} \right\rfloor = 0
\end{align*}
\]

D. When transmitting the information about EECS majors over a noisy communication link, the department has chosen to use the 7-bit encoding shown on the right in the hopes that they'll be able to correct multiple-bit errors during transmission. Using this code, how many bit errors in a message about a single major will the receiver be able to correct?

\[
\text{correct} = \left\lfloor \frac{HD - 1}{2} \right\rfloor = 1 \text{ bit error correction}
\]
E. We wish to transmit messages comprised of the four symbols shown below with their associated probabilities and 5-bit fixed-length encoding

<table>
<thead>
<tr>
<th>Symbol</th>
<th>p(symbol)</th>
<th>encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
<td>0.5</td>
<td>00000</td>
</tr>
<tr>
<td>β</td>
<td>0.125</td>
<td>11100</td>
</tr>
<tr>
<td>γ</td>
<td>0.25</td>
<td>11011</td>
</tr>
<tr>
<td>δ</td>
<td>0.125</td>
<td>10111</td>
</tr>
</tbody>
</table>

If we transmit messages using the 5-bit fixed-length encoding shown above, will it be possible to perform single-bit error detection and correction at the receiver?

\[
\text{correction: } \left\lfloor \frac{\text{HD} - 1}{2} \right\rfloor = 0 \implies \text{No!}
\]

F. What is the Hamming distance between the encodings for A and B?

Using an encoding scheme with this Hamming distance, how many bits of error can be detected? How many bits of error can be corrected?

\[
\text{A: 010010} \quad \text{B: 110161}
\]

\[
\text{HD} = 4 \\
\text{detected} = \text{HD} - 1 = 3 \\
\text{corrected} = \left\lfloor \frac{\text{HD} - 1}{2} \right\rfloor = 1
\]
An internet Sudoku gaming site transmits messages containing nine data bits and seven parity bits, arranged in a rectangle as follows:

<table>
<thead>
<tr>
<th>D₀₀</th>
<th>D₀₁</th>
<th>D₀₂</th>
<th>P₀ₓ</th>
</tr>
</thead>
<tbody>
<tr>
<td>D₁₀</td>
<td>D₁₁</td>
<td>D₁₂</td>
<td>P₁ₓ</td>
</tr>
<tr>
<td>D₂₀</td>
<td>D₂₁</td>
<td>D₂₂</td>
<td>P₂ₓ</td>
</tr>
<tr>
<td>Pₓ₀</td>
<td>Pₓ₁</td>
<td>Pₓ₂</td>
<td>Pₓₓ</td>
</tr>
</tbody>
</table>

Each Dᵢ in the above diagram indicates a data bit, equally likely to be a 0 or 1. Each Pᵢₓ and Pₓᵢ is an odd parity bit chosen to make the total number of 1s in the iᵗʰ row or jᵗʰ column, respectively, odd. Pₓₓ is an odd parity bit chosen to make the total number of 1s in the entire transmission odd. Thus in an error-free transmission, the total number of 1s in 4-bit columns 0 thru 2 and 4-bit rows 0 thru 2, as well as in the entire 16-bit transmission, is odd.

Note that each 9-bit data word determines a unique 16-bit valid codeword to be transmitted.

G. What is the minimum Hamming distance between valid codewords? [Hint: flipping one bit of the data word changes how many bits of the codeword?]

| Flipped data bit ⇒ flip row & col parity |
| 3 bit have changed so flip overall parity |

Each of the following represents a transmission received, with at most a single-bit error. For each message, circle the bit, if any, that was changed due to a transmission error.

H.

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

I.

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

J.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

K.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

check row, col, overall parity
16 of 1s should be odd

6.004 Worksheet - 12 of 12 - L01 - Basics of Information