Show All Answers Hide All Answers

6.02 Practice Problems: Information, Entropy, & Source Coding

---

Problem 1. Huffman coding is used to compactly encode the species of fish tagged by a game warden. If 50% of the fish are bass and the rest are evenly divided among 15 other species, how many bits would be used to encode the species when a bass is tagged?

Hide Answer

If a symbol has a probability of ≥ .5, it will be incorporated into the Huffman tree on the final step of the algorithm, and will become a child of the final root of the decoding tree. This means it will have a 1-bit encoding.

Problem 2. Several people at a party are trying to guess a 3-bit binary number. Alice is told that the number is odd; Bob is told that it is not a multiple of 3 (i.e., not 0, 3, or 6); Charlie is told that the number contains exactly two 1's; and Deb is given all three of these clues. How much information (in bits) did each player get about the number?

Hide Answer

N = 8 choices for a 3-bit number
Alice: odd = {001, 011, 101, 111} M=4, info = log2(8/4) = 1 bit
Bob: not a multiple of 3 = {001, 010, 100, 101, 111} M=5, info = log2(8/5) = .6781 bits
Charlie: two 1's = {011, 101, 110} M=3, info = log2(8/3) = 1.4150
Deb: odd, not a multiple of 3, two 1's = {101}, M=1, info = log(8/1) = 3 bits

---

Problem 3. X is an unknown 8-bit binary number. You are given another 8-bit binary number, Y, and told that Y differs from X in exactly one bit position. How many bits of information about X have you been given?

Hide Answer

Knowing Y, we've narrowed down the value of X to one of 8 choices. Bits of information = log2(256/8) = 5 bits.

Problem 4. In Blackjack the dealer starts by dealing 2 cards each to himself and his opponent: one face down, one face up. After you look at your face-down card, you know a total of three cards. Assuming this was the first hand played from a new deck, how many bits of information do you now have about the dealer's face down card?

Hide Answer

We've narrowed down the choices for the dealer's face-down card from 52 (any card in the deck) to one of 49 cards (since we know it can't be one of three visible cards. bits of information = log2(52/49).

Problem 5. The following table shows the current undergraduate and MEng enrollments for the School of Engineering (SOE).

Course (Department)	# of students	Prob.
I (Civil & Env.)	121	.07
II (Mech. Eng.)	389	.23
III (Mat. Sci.)	127	.07
VI (EECS)	645	.38
X (Chem. Eng.)	237	.13
XVI (Aero & Astro)	198	.12
Total	1717	1.0

A. When you learn a randomly chosen SOE student's department you get some number of bits of information. For which student department do you get the *least* amount of information?

Hide Answer

We'll get the smallest value for log2(1/p) by choosing the choice with the largest p => EECS.

B. Design a variable length Huffman code that minimizes the average number of bits in messages encoding the departments of randomly chosen groups of students. Show your Huffman tree and give the code for each course.

Hide Answer

step 1 = {(I,.07) (II,.23) (III,.07) (VI,.38) (X,.13) (XVI,.12)}
step 2 = {([I,III],.14) (II,.23) (VI,.38) (X,.13) (XVI,.12)}
step 3 = {([I,III],.14) (II,.23) (VI,.38) ([X,XVI],.25)}
step 4 = {([II,[I,III]],.37) (VI,.38) ([X,XVI],.25)}
step 5 = {([[X,XVI],[II,[I,III]]],.62) (VI,.38)}
step 6 = {([[[X,XVI,[II,[I,III]],VI],1.0)}

code for course I: 0 1 1 0
code for course II: 0 1 0
code for course III: 0 1 1 1
code for course VI: 1
code for course X: 0 0 0
code for course XVI: 0 0 1

There are of course many equivalent codes derived by swapping the "0" and "1" labels for the children of any interior node of the decoding tree. So any code that meets the following constraints would be fine:

code for VI = length 1
code for II, X, XVI = length 3
code for I, III = length 4

C. If your code is used to send messages containing only the encodings of the departments for each student in groups of 100 randomly chosen students, what's the average length of such messages? Write an expression if you wish.

Hide Answer

100*[(.38)(1) + (.23 + .13 + .12)*(3) + (.07 + .07)*(4)] = 238 bits

---

Problem 6. You're playing an on-line card game that uses a deck of 100 cards containing 3 Aces, 7 Kings, 25 Queens, 31 Jacks and 34 Tens. In each round of the game the cards are shuffled, you make a bet about what type of card will be drawn, then a single card is drawn and the winners are paid off. The drawn card is reinserted into the deck before the next round begins.

A. How much information do you receive when told that a Queen has been drawn during the current round?

Hide Answer

information received = log2(1/p(Queen)) = log2(1/.25) = 2 bits

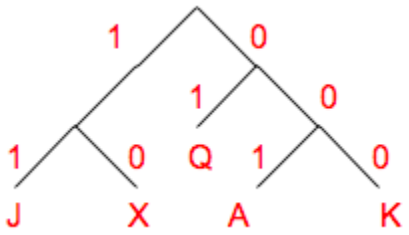
B. Give a numeric expression for the average information content received when learning about the outcome of a round (aka the entropy).

Hide Answer

(.03)log2(1/.03) + (.07)log2(1/.07) + (.25)log2(1/.25) + (.31)log2(1/.31) + (.34)log2(1/.34) = 1.97 bits

C. Construct a variable-length Huffman encoding that minimizes the length of messages that report the outcome of a sequence of rounds. The outcome of a single round is encoded as A (ace), K (king), Q (queen), J (jack) or X (ten). Specify your encoding for each of A, K, Q, J and X.

Hide Answer



Encoding for A: 001
Encoding for K: 000
Encoding for Q: 01
Encoding for J: 11
Encoding for X: 10

D. Using your code from part (C) what is the expected length of a message reporting the outcome of 1000 rounds (i.e., a message that contains 1000 symbols)?

Hide Answer

(.07+.03)(3 bits) + (.25+.31+.34)(2 bits) = 2.1 bits average symbol length using Huffman code. So expected length of 1000-symbol message is 2100 bits.

E. The Nevada Gaming Commission regularly receives messages in which the outcome for each round is encoded using the symbols A, K, Q, J and X. They discover that a large number of messages describing the outcome of 1000 rounds (i.e. messages with 1000 symbols) can be compressed by the LZW

algorithm into files each containing 43 bytes in total. They immediately issue an indictment for running a crooked game. Briefly explain their reasoning.

Hide Answer

43 bytes is 344 bits, way below the entropy of 1970 bits for encoding 1000 symbols. This indicates the actual symbol probabilities must be dramatically different than the stated probabilities that were used to calculate the entropy in part (B). The experimentally determined value of the actual entropy (344/1000 = .344 bits) indicates that one or more of the symbols must have a much higher probability of occuring than stated, which suggests a rigged game.

Problem 7.

Consider messages made up entirely of vowels (A, E, I, O, U). Here's a table of probabilities for each of the vowels:

l	p(l)	log2(1/p(l))	p(l)*log2(1/p(l))
A	0.22	2.18	0.48
E	0.34	1.55	0.53
I	0.17	2.57	0.43
O	0.19	2.40	0.46
U	0.08	3.64	0.29
Totals	1.00	12.34	2.19

A. Give an expression for the number of bits of information you receive when learning that a particular vowel is either I or U.

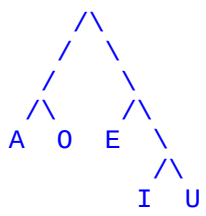
Hide Answer

The probability that a letter is either I or U is p(I) + p(U) = 0.25. So the number of bits of information you receive is log2(1/.25) = 2 bits.

B. Using Huffman's algorithm, construct a variable-length code assuming that each vowel is encoded individually. Please draw a diagram of the Huffman tree and give the encoding for each of the vowels.

Hide Answer

Running Huffman's algorithm:



So assuming we label left branches with "0" and right branches with "1", the encodings are

A = 00
O = 01
E = 10
I = 110

U = 111

C. Using your code from above, give an expression for the expected length in bits of an encoded message transmitting 100 vowels.

Hide Answer

Expected length for one vowel = Σ p(l)*len(encoding(l)):

0.22*2 + 0.19*2 + 0.34*2 + 0.17*3 + 0.08*3 = 2.25 bits

So the expected length for 100 vowels is 225 bits.

D. Ben Bitdiddle spends all night working on a more complicated encoding algorithm and sends you email claiming that using his code the expected length in bits of an encoded message transmitting 100 vowels is 197 bits. Would you pay good money for his implementation?

Hide Answer

No! The entropy is 2.19 bits which means that 219 bits is a lower bound on the number of bits needed on the average to encode 100 vowels. Ben's encoding uses less bits, which is impossible, so his code or his derviation of the expected lenth must be bogus in some way.

Problem 8. Describe the contents of the string table created when encoding a very long string of all a's using the simple version of the LZW encoder shown below. In this example, if the decoder has received E encoded symbols (i.e., string table indices) from the encoder, how many a's has it been able to decode?

```
initialize TABLE[0 to 255] = code for individual bytes
STRING = get input symbol
while there are still input symbols:
    SYMBOL = get input symbol
    if STRING + SYMBOL is in TABLE:
        STRING = STRING + SYMBOL
    else:
        output the code for STRING
        add STRING + SYMBOL to TABLE
        STRING = SYMBOL
output the code for STRING
```

Hide Answer

The string table is filled with sequences of a's, each one longer than the last. So

table[256] = aa
table[257] = aaa
table[258] = aaaa
...

Here's what the decoder receives, and the decoding it does (it's instructive to figure out how the decoder is building its copy of the string table based on what it's receiving):

index of a => decode a
256 => decode aa
257 => decode aaa

258 => decode aaaa
...

So if E symbols have arrived, the decoder has produced 1 + 2 + 3 + ... + E a's, which sums to E(E+1)/2.

Problem 9. Consider the pseudo-code for the LZW decoder given below:

```
initialize TABLE[0 to 255] = code for individual bytes
CODE = read next code from encoder
STRING = TABLE[CODE]
output STRING

while there are still codes to receive:
    CODE = read next code from encoder
    if TABLE[CODE] is not defined:
        ENTRY = STRING + STRING[0]
    else:
        ENTRY = TABLE[CODE]
    output ENTRY
    add STRING+ENTRY[0] to TABLE
    STRING = ENTRY
```

Suppose that this decoder has received the following five codes from the LZW encoder (these are the first five codes from a longer compression run):

```
97 -- index of 'a' in the translation table
98 -- index of 'b' in the translation table
257 -- index of second addition to the translation table
256 -- index of first addition to the translation table
258 -- index of third addition to the translation table
```

After it has finished processing the fifth code, what are the entries in TABLE and what is the cumulative output of the decoder?

Hide Answer

Here's the step-by-step operation of the decoder:

CODE=97, STRING='a', output 'a'
CODE=98, ENTRY='b', output 'b', TABLE[256]='ab', STRING='b'
CODE=257, ENTRY='bb', output 'bb', TABLE[257]='bb', STRING='bb'
CODE=256, ENTRY='ab', output 'ab', TABLE[258]='bba', STRING='ab'
CODE=258, ENTRY='bba', output 'bba', TABLE[259]='abb', STRING='bba'

So the cummulative output of the decoder is 'abbbabbba'.

Problem 10. Huffman and other coding schemes tend to devote more bits to the coding of

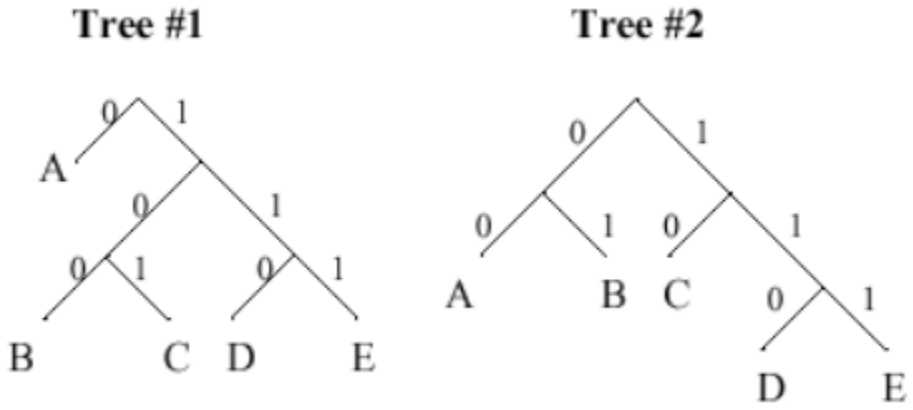
(A) symbols carrying the most information
(B) symbols carrying the least information
(C) symbols that are likely to be repeated consecutively
(D) symbols containing redundant information

Hide Answer

Answer is (A). Symbols carrying the most information, i.e., the symbols that are less likely to occur. This

makes sense: to keep messages as short as possible, frequently occurring symbols should be encoded with fewer bits and infrequent symbols with more bits.

Problem 11. Consider the following two Huffman decoding tress for a variable-length code involving 5 symbols: A, B, C, D and E.



A. Using Tree #1, decode the following encoded message: "01000111101".

Hide Answer

Starting at the root of the decoding tree, use the bits of the message to traverse the tree until a leaf node is reached; output that symbol. Repeat until all the bits in the message are consumed.

0 = A
100 = B
0 = A
111 = E
101 = C

B. Suppose we were encoding messages with the following probabilities for each of the 5 symbols: p(A) = 0.5, p(B) = p(C) = p(D) = p(E) = 0.125. Which of the two encodings above (Tree #1 or Tree #2) would yield the shortest encoded messages averaged over many messages?

Hide Answer

Using Tree #1, the expected length of the encoding for one symbol is:

1*p(A) + 3*p(B) + 3*p(C) + 3*p(D) + 3*p(E) = 2.0

Using Tree #2, the expected length of the encoding for one symbol is:

2*p(A) + 2*p(B) + 2*p(C) + 3*p(D) + 3*p(E) = 2.25

So using the encoding represented by Tree #1 would yield shorter messages on the average.

C. Using the probabilities of part (B), if you learn that the first symbol in a message is "B", how many bits of information have you received?

Hide Answer

bits of info received = log2(1/p) = log2(1/.125) = 3

D. Using the probabilities of part (B), If Tree #2 is used to encode messages what is the average length of 100-symbol messages, averaged over many messages?

Hide Answer

In part (B) we calculated that the expected length of the encoding for one symbol using Tree #2 was 2.25 bits, so for 100-symbol messages the expected length is 225 bits.

Problem 12. Ben Bitdiddle has been hired by the Registrar to help redesign the grades database for WebSIS. Ben is told that the database only needs to store one of five possible grades (A, B, C, D, F). A survey of the current WebSIS repository reveals the following probabilities for each grade:

Grade	Probability of occurrence
A	p(A) = 18%
B	p(B) = 27%
C	p(C) = 25%
D	p(D) = 15%
F	p(E) = 15%

A. Given the probabilities above, if you are told that a particular grade is "C", give an expression for the number of bits of information you have received.

Hide Answer

# of bits for "C" = log(1/pC) = log(1/.25) = log(4) = 2

B. Ben is interested in storing a sequence of grades using as few bits as possible. Help him out by creating a variable-length encoding that minimizes the average number of bits stored for a sequence of grades. Use the table above to determine how often a particular grade appears in the sequence.

Hide Answer

using Huffman algorithm:
- since D,F are least probable, make a subtree of them, p(D/\F) = 30%
- now A,C are least probable, make a subtree of them, P(A/\C) = 43%
- now B,DF are least probable, make a subtree of them P(B/\(D/\F)) = 55%
- just AC,BDF are left, make a subtree of them (A/\C)/\(B/\(D/\F))
- so A = 00, B = 10, C = 01, D = 110, F = 111

Problem 13. Consider a sigma-delta modulator used to convert a particular analog waveform into a sequence of 2-bit values. Building a histogram from the 2-bit values we get the following information:

Modulator value	# of occurrences
00	25875
01	167836

10	167540
11	25974

A. Using probabilities computed from the histogram, construct a variable-length Huffman code for encoding these four values.

Hide Answer

p(00)=.0668 p(01)=.4334 p(10)=.4327 p(11)=.0671
Huffman tree = 01 /\ (10 /\ (00 /\ 11))
Code: 01="0" 10="10" 00="110" 11="111"

B. If we transmit the 2-bit values as is, it takes 2 bits to send each value (doh!). If we use the Huffman code from part (A) what is the average number of bits used to send a value? What compression ratio do we achieve by using the Huffman code?

Hide Answer

sum(pi*len(code for pi)) = 1.7005, which is 85% of the original 2-bit/symbol encoding.

C. Using Shannon's entropy formula, what is the average information content associated with each of the 2-bit values output by the modulator? How does this compare to the answers for part (B)?

Hide Answer

sum(pi*log2(1/pi)) = 1.568, so the code of part(B) isn't quite as efficient as we can achieve by using an encoding that codes multiple pairs as in one code symbol.

Problem 14. In honor of Daisuke Matsuzaka's first game pitching for the Redsox, the Boston-based members of the Search for Extraterrestrial Intelligence (SETI) have decided to broadcast a 1,000,000 character message made up of the letters "R", "E", "D", "S", "O", "X". The characters are chosen at random according the probabilities given in the table below:

Letter	p(Letter)
R	.21
E	.31
D	.11
S	.16
O	.19
X	.02

A. If you learn that one of the randomly chosen letters is a vowel (i.e., "E" or "O") how many bits of information have you received?

Hide Answer

p(E or O) = .31 + .19 = .5, log2(1/pi) = 1 bit

B. Nervous about the electric bill for the transmitter at Arecibo, the organizers have asked you to design a variable length code that will minimize the number of bits needed to send the message of 1,000,000 randomly chosen characters. Please draw a Huffman decoding tree for your code, clearly labeling each leaf with the appropriate letter and each arc with either a "0" or a "1".

Hide Answer

Huffman algorithm:
choose X,D: X/\D, p = .13
choose S,XD: S/\(X/\D), p = .29
choose R,0: R/\O, p = .40
choose E,SXD: E/\(S/\(X/\D)), p = .6
choose RO,ESXD: (R/\O) /\ (E/\(S/\(X/\D)))
code: R=00 O=01 E=10 S=110 X=1110 D=1111

C. Using your code, what bit string would they send for "REDSOX"?

Hide Answer

00 10 1111 110 01 1110

Problem 15. "Information, please"

A. You're given a standard deck of 52 playing cards that you start to turn face up, card by card. So far as you know, they're in completely random order. How many new bits of information do you get when the first car is flipped over? The fifth card? The last card?

Hide Answer

First card: log2(52/1)

Fifth card: log2(48/1)

Last card: log2(1/1) = 0 (no information since, knowing the other 51 cards, one can predict with certainty the last card)

B. Suppose there three alternatives ("A", "B" and "C") with the following probabilities of being chosen:

p("A") = 0.8
p("B") = 0.1
p("C") = 0.1

We might encode the of "A" with the bit string "0", the choice of "B" with the bit string "10" and the choice of "C" with the bit string "11".

If we record the results of making a sequence of choices by concatenating in left-to-right order the bit strings that encode each choice, what sequence of choices is represented by the bit string "00101001100000"?

Hide Answer

AABBACAAAAA

C. Using the encoding of the previous part, what is the expected length of the bit string that encodes the results of making 1000 choices? What is the length in the worst case? How do these numbers compare with 1000*log2(3/1), which is the information content of 1000 *equally-probable* choices?

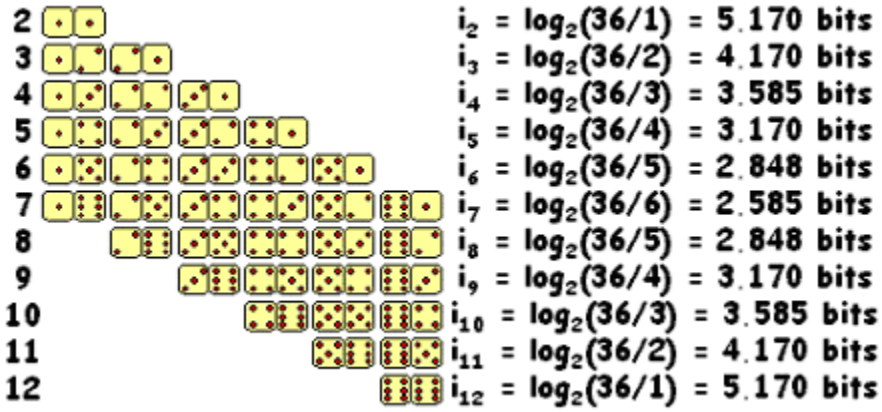
Hide Answer

Expected length = 1000[(0.8)(1) + (0.1)(2) + (0.1)(2)] = 1200 bits

In the worst case, each choice would take 2 bits to encode = 2000 bits.

1000*log2(3/1) = 1585. In general, a variable-length encoding of sequences of N choices with differing probabilities gets better compression than can be achieved if the choices are equally probable.

D. Consider the sum of two six-sided dice. Even when the dice are "fair" the amount information conveyed by a single sum depends on what the sum is since some sums are more likely than others, as shown in the following figure:



What is the average number of bits of information provided by the sum of 2 dice? Suppose we want to transmit the sums resulting from rolling the dice 1000 times. How many bits should we expect that transmission to take?

Hide Answer

Average number of bits = sum (p_i)log2(1/p_i) for i = 2 through 12. Using the probabilities given in the figure above the average number of bits of information provided by the sum of two dice is 3.2744.

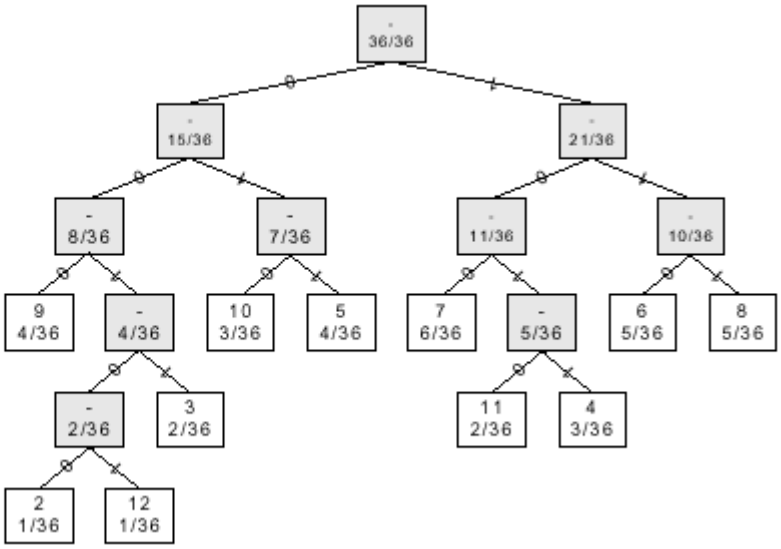
So if we had the perfect encoding, the expected length of the transmission would be 3274.4 bits. If we encode each sum separately we can't quite achieve this lower bound -- see the next question for details.

E. Suppose we want to transmit the sums resulting from rolling the dice 1000 times. If we use 4 bits to encode each sum, we'll need 4000 bits to transmit the result of 1000 rolls. If we use a variable-length binary code which uses shorter sequences to encode more likely sums then the expected number of bits need to encode 1000 sums should be less than 4000. Construct a variable-length encoding for the sum of two dice whose expected number of bits per sum is less than 3.5. (Hint: It's possible to find an encoding for the sum of two dice with an expected number of bits = 3.306.)

Hide Answer

Using Huffman's algorithm, we arrive at the following encoding which has 3.3056 as the expected number of bits for each sum.

Sum	Prob	Encoding
2	1/36	00100
3	2/36	0011
4	3/36	1011
5	4/36	011
6	5/36	110
7	6/36	100
8	5/36	111
9	4/36	000
10	3/36	010
11	2/36	1010
12	1/36	00101



F. Can we make an encoding for transmitting 1000 sums that has an expected length smaller than that achieved by the previous part?

Hide Answer

Yes, but we have to look at encoding more than one sum at a time, e.g., by applying the construction algorithm to pairs of sums, or ultimately to all 1000 sums at once. Many of the more sophisticated compression algorithms consider sequences of symbols when constructing the appropriate encoding scheme.

Problem 16.

Consider messages comprised of four symbols -- A, B, C, D -- each with an associated probability of occurrence: p(A), p(B), p(C), p(D). Suppose $p(A) \geq p(B) \geq p(C) \geq p(D)$. Write down a single condition (equation or inequality) that is both necessary and sufficient to guarantee that the Huffman algorithm will generate a two-bit encoding for each symbol, i.e., the corresponding Huffman code will actually be a fixed-length encoding using 2 bits for each symbol.

Hide Answer

$p(C) + p(D) > p(A)$. The reason is that there are only two possible Huffman trees with 4 leaves. For the tree where every symbol is encoded with two bits, we need to have $p(C) + p(D) + p(B) > p(B) + p(A)$ to ensure that the algorithm will combine B with A instead of with C/D. We need a strict inequality because otherwise there is a possiblity of a tie and the unbalanced tree may be chosen since it has the same expected number of bits.

Problem 17. Consider a Huffman code over four symbols: A, B, C, and D. For each of the following encodings indicate if it is a valid Huffman code, i.e., a code that would have been produced by the Huffman algorithm given some set of probabilities p(A), ..., p(D).

A. A:0, B:11, C:101, D:100

Hide Answer

Valid. This is one of the two possible Huffman trees with 4 leaves.

B. A: 1, B:01, C:00, D:010

Hide Answer

Not valid. This code can't be unambigously decoded by the receiver since the encoding for B is a prefix for the encoding of D. For example, the message 0101 might be decoded as "BB" or "DA" -- the receiver doesn't have sufficient information to tell which.

C. A:00, B:01, C:110, D:111

Hide Answer

Not valid. This a legal variable length code, but it's not optimal. A shorter code would have C and D encoded in 2 bits, as 10 and 11 (or vice versa), and that would be a Huffman code for the same symbol probabilities, not the one given.

Problem 18.

After careful data collection, Alyssa P. Hacker observes that the probability of HIGH or LOW traffic on Storrow Drive is given by the following table:

Condition	p(HIGH traffic)	p(LOW traffic)
If the Redsox are playing	0.999	.001
If the Redsox are not playing	0.25	0.75

A. If it is known that the Red Sox are playing, then how many bits of information are conveyed by the statement that the traffic level is LOW.

Hide Answer

log2(1/0.001) = 9.966 bits.

B. Suppose it is known that the Red Sox are *not* playing. What is the entropy of the corresponding probability distribution of traffic?

Hide Answer

0.25*log2(1/0.25) + 0.75*log2(1/0.75) = 0.811 bits.

Problem 19.

Consider Huffman coding over four symbols (A, B, C and D) with probabilities p(A)=1/3, p(B)=1/2, p(C)=1/12 and p(D)=1/12.

The entropy of the discrete random variable with this probability distribution was calculated to be 1.62581 bits.

We then used the Huffman algorithm to build the following variable length code:

A: 10
B: 0

C: 110
D: 111

which gave an expected encoding length of 1.666 bits/symbol, slightly higher than the entropy bound.

A. Suppose we made up a new symbol alphabet consisting of all possible pairs of the original four symbols. Enumerate the new symbols and give the probabilities associated with each new symbol.

Hide Answer

Here are the 16 new symbols and their associated probabilities:

```
AA  p(AA)=1/9
AB  p(AB)=1/6
AC  p(AC)=1/36
AD  p(AD)=1/36
BA  p(BA)=1/6
BB  p(BB)=1/4
BC  p(BC)=1/24
BD  p(BD)=1/24
CA  p(CA)=1/36
CB  p(CB)=1/24
CC  p(CC)=1/144
CD  p(CD)=1/144
DA  p(DA)=1/36
DB  p(DB)=1/24
DC  p(DC)=1/144
DD  p(DD)=1/144
```

B. What is the entropy associated with the discrete random variable that has the probability distribution you gave in part (A). Is is the same, bigger, or smaller than the entropy of 1.626 calculated above? Explain.

Hide Answer

The entropy for the new double-symbol alphabet is 3.25163 bits/double-symbol, exactly twice that of the entropy of the original alphabet. This means that we haven't changed the expected information content by choosing to encode pairs of symbols instead of one symbol at a time.

C. Derive the Huffman encoding for the new symbols and compute the expected encoding length expressed in bits/symbol. How does it compare with the 1.666 bits/symbol for the original alphabet? Note: this is tedious to do by hand -- try using the Huffman program you wrote!

Hide Answer

```
AB = 000
BA = 001
BB = 01
DB = 10000
DD = 1000100
CC = 1000101
CD = 1000110
DC = 1000111
AD = 10010
CA = 10011
AA = 101
DA = 11000
AC = 11001
BC = 1101
```

```
BD = 1110
CB = 1111
```

The expected length of encoding a choice is 3.29167 bits/double-symbol, which is slightly less than twice 1.6666 bits/symbols. In other words, the expected encoded length of a message will be slightly smaller if we encode pairs of symbols using the encoding above than if we used the original one-symbol-at-a-time encoding.

6.02 Introduction to EECS II: Digital Communication Systems

Fall 2012