50.002 COMPUTATIONAL STRUCTURES

INFORMATION SYSTEMS TECHNOLOGY AND DESIGN

Problem Set 1

1 Measuring Information

1. Someone picks a name out of a hat known to contain the names of 5 women and 3 men, and tells you a man has been selected. How much information have they given you about the selection?

   Solution:
   Originally, we have 8 options: $M = 8$. The option is narrowed down into 3, hence $N = 3$, we have:

   $$I = \log_2\left(\frac{1}{N/M}\right) = \log_2\left(\frac{1}{3/8}\right) \text{ bits of information.} \tag{1}$$

2. 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 card is flipped over? The fifth card? The last card?

   Solution:
   The first card has $\log_2 52$ bits of information. The fifth card has $\log_2 48$ bits of information. The last card has 0 bits of information.

3. X is an unknown N-bit binary number ($N > 3$). You are told that the first three bits of X are 011. How many bits of information have you been given?

   Solution:
   The bits of information that has been given is 3.

4. X is an unknown 8-bit binary number. You are given another 8-bit binary number, Y, and told that the Hamming distance (number of different bits) between X and Y is one. How many bits of information about X have you been given when Y is presented to you?

Solution:
Originally, we have 8 unknown bits, that is $M = 2^8$ choices. After given Y, we are down to $N = 8$ choices, each having 1 bitt different for X. Therefore, we have been given,

$$I = \log_2\left(\frac{1}{N/M}\right) = \log_2\left(\frac{1}{8/2^8}\right) = 5 \text{ bits of information.} \tag{2}$$

2 Measuring Information

After spending the afternoon in the dentist's chair, Ben has invented a new language called DDS made up entirely of vowels (the only sounds he could make with someone's hand in his mouth). The DDS alphabet consists of the five letters: A, E, I, U, O, which occur with the following probabilities,

Letter	Prob. of occurrence
A	$p(A) = 0.15$
E	$p(B) = 0.4$
I	$p(C) = 0.15$
O	$p(D) = 0.15$
U	$p(E) = 0.15$

Table 1

If you're told that the first letter of the message is "A", give an expression for the number of bits of information you have received.

Solution:
Recall that the information received is inversely proportional to the probability of that choice occurring, and we take $\log_2$ of the probability of that choice occurring to quantify the information in terms of bits. Hence the expression is,

$$I = \log_2 \frac{1}{p(A)} \tag{3}$$

3 Modular arithmetic and 2's complement representation

Most computers choose a particular word length (measured in bits) for representing integers and provide hardware that performs various arithmetic operations on word-size operands. The current generation of processors have word lengths of 32 bits; restricting the size of the operands and the result to a single word means that the arithmetic operations are actually performing arithmetic modulo 2^{32}.

Almost all computers use a 2's complement representation for integers since the 2's complement addition operation is the same for both positive and negative numbers. In 2's complement notation, one negates a number by forming the 1's complement (i.e: for each bit, changing 0 to a 1 and vice versa) representation of the number and then adding 1. **By convention**, we write 2's complement integers with the most-significant bit (MSB) on the left and the least-significant bit (LSB) on the right. Also, **by convention**, if the MSB is 1, the number is negative, otherwise it's non-negative.

1. How many different values can be encoded in a 32-bit word?
   Solution:
   2^{32}.

2. Please use a 32-bit 2's complement representation to answer the following questions. What are the representations for:

   (a) Zero
   Solution:
   `0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0`

   (b) The most positive integer that can be represented
   Solution:
   `0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1`

   (c) The most negative integer that can be represented
   Solution:
   `1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0`

   What are the decimal values for the most positive and the most negative number?
   Solution:
   The most positive value in decimal: 2147483647. The most negative value in decimal: -2147483648

3. Since writing a string of 32 bits gets tedious, it's often convenient to use hexadecimal representation where a single digit in the range of 0-9 or A-F is used to represent groups of 4 bits. Give the 8-digit hexadecimal equivalent of the following decimal and binary numbers:

   (a) 37_{10}
   Solution:
   `0x 0000 0025`

   (b) -32768_{10}
   Solution:
   We begin by converting 32768 (positive) number to Hex: `0x 0000 8000`. Then we take the 1's complement: `0x FFFF 7FFF`, and finally $+1$: `0x FFFF 8000`.

(c) `1101 1110 1010 1101 1011 1110 1110 1111`
Solution:
`0x DEAD BEEF`

4. Calculate the following using 6-bit 2's complement arithmetic (which is just a fancy way of saying to do ordinary addition in base 2, keeping only 6 bits of your answer). Show your work using binary notation. Remember that subtraction can be performed by negating the second operand and then adding to the first operand.

(a) 13 + 10
Solution:
`001101 + 001010 = 010111`

(b) 15 - 18
Solution:
18 is `010010`. -18 is `101110`. Hence, `001111 + 101110 = 111101`

(c) 27 - 6
Solution:
`011011 + 111010 = 010101`

5. At first blush, "Complement and add 1" doesn't seem like an obvious way to negate a two's complement number. By manipulating the expression $A+(-A)=0$, show that "complement ad add 1" does produce correct representation for the negative of a two's complement number.
Hint: express 0 as (-1 + 1) and rearrange the terms to get -A on one side and XXX+1 on the other and then think about how the expression XXX is related to A using only logical operations (AND, OR, NOT).

$$\begin{aligned}(-A) &= -A - 1 + 1 \qquad (4)\\ &= (-1 - A) + 1 \qquad (5)\end{aligned}$$

In this case, XXX is (-1 - A). Let's say we have 8 bit number. -1 is represented by all 1's : `1111 1111`. Let's represent A arbitrarily as `a7 a6 a5 a4 a3 a2 a1 a0`, where a_i can be 0 or 1.

Subtracting -1 with A will flip the bits of A, such that if $a_i = 0$, then $1 - a_i = 1$, and if $a_i = 1$ then $1 - a_i = 0$ (see how binary subtraction 'borrow' method works at `https://www.wikihow.com/Subtract-Binary-Numbers` if you dont know how it works). Hence, we can rewrite the above into,

$$(-A) = \textit{bitwise}\ \text{NOT}\ A + 1 \qquad (6)$$