September 14, 2003

Bitstream exercises in class

Background assumptions Our messages in bitstreams will be a succession of words which will be one of three types (the given percentage frequencies are approximate):

- 50% of the words are 1010101010. This is 10 bits long, alternating 1's and 0's. Call it **P**.
- 25% of the words are 1111111. This is 7 bits long, all 1's. Call it **Q**.
- 25% of the words are 000000. This is 6 bits long, all 0's. Call it **R**.

We'll call bitstreams composed of these words, sentences. The sentence **RQPR** is:

`00000011111111010101010000000`

Bitstreams will divided into groups of five bits each to make it easier to refer to parts of them. Then the sentence above is written:

`00000 01111 11110 10101 01000 0000`

Problem 1

Write the following sentence in terms of **P**, **Q**, and **R**.

`10101 01010 11111 11101 01010 10000 00010 10101 01011 11111 10101 01010`

Problem 2

Here is a sentence which has been xored with a pseudorandom bitstream with approximately 10% 1's. Write the original sentence in terms of **P**, **Q**, and **R**.

`11011 11101 01010 10101 00010 10001 01010 00010 00010 10101 01011 11011`

Problem 3

The following bitstream is two sentences xored together. Write the original sentences in terms of **P**, **Q**, and **R** as well as you can.

`01010 10111 10101 01000 00000 00101 01000 00101 01001 01010 11110 10101`

Problem 4

A pseudorandom bitstream with approximately 50% 1's and 50% 0's has been created. Two <u>different</u> sentences have been xored with it and the results are shown. Write the original sentences in terms of **P**, **Q**, and **R** as well as you can.

`11110 00111 00101 10010 10010 10011 11101 11110 00100 10111 01110 11011 100`

`11110 00111 10000 10010 11000 00110 10111 10100 10101 11101 00100 01001 00`

Some statistics about English

The web page
`http://www.cmb.ac.lk/academic/Science/Computer/dscs/courses/Computer/Msc/DSandC/english.htm`
contains the following table, which the web page quotes from *Computer Networks* by A. S. Tanenbaum (Prentice-Hall, 1989).

Letters		Bigrams		Trigrams		Words	
E	13.05	TH	3.16	THE	4.72	THE	6.42
T	9.02	IN	1.54	ING	1.42	OF	4.02
O	8.21	ER	1.33	AND	1.13	AND	3.15
A	7.81	RE	1.30	ION	1.00	TO	2.36
N	7.28	AN	1.08	ENT	0.98	A	2.09
I	6.77	HE	1.08	FOR	0.76	IN	1.77
R	6.64	AR	1.02	TIO	0.75	THAT	1.25
S	6.46	EN	1.02	ERE	0.69	IS	1.03
H	5.85	TI	1.02	HER	0.68	I	0.94
D	4.11	TE	0.98	ATE	0.66	IT	0.93
L	3.60	AT	0.88	VER	0.63	FOR	0.77
C	2.93	ON	0.84	TER	0.62	AS	0.76
F	2.88	HA	0.84	THA	0.62	WITH	0.76
U	2.77	OU	0.72	ATI	0.59	WAS	0.72
M	2.62	IT	0.71	HAT	0.55	HIS	0.71
P	2.15	ES	0.69	ERS	0.54	HE	0.71
Y	1.51	ST	0.68	HIS	0.52	BE	0.63
W	1.49	OR	0.68	RES	0.50	NOT	0.61
G	1.39	NT	0.67	ILL	0.47	BY	0.57
B	1.28	HI	0.66	ARE	0.46	BUT	0.56
V	1.00	EA	0.64	CON	0.45	HAVE	0.55
K	0.42	VE	0.64	NCE	0.45	YOU	0.55
X	0.30	CO	0.59	ALL	0.44	WHICH	0.53
J	0.23	DE	0.55	EVE	0.44	ARE	0.50
Q	0.14	RA	0.55	ITH	0.44	ON	0.47
Z	0.09	RO	0.55	TED	0.44	OR	0.45

The web page cited has many other numbers about English which you might find interesting. There are also many other references for English and other languages.

September 14, 2003

Answers to the bitstream problems in class

Background assumptions Our messages in bitstreams will be a succession of words which will be one of three types (the given percentage frequencies are approximate):

- 50% of the words are 1010101010. This is 10 bits long, alternating 1's and 0's. Call it **P**.
- 25% of the words are 1111111. This is 7 bits long, all 1's. Call it **Q**.
- 25% of the words are 000000. This is 6 bits long, all 0's. Call it **R**.

We'll call bitstreams composed of these words, sentences. The sentence **RQPR** is:

`00000011111111010101010000000`

Bitstreams will divided into groups of five bits each to make it easier to refer to parts of them. Then the sentence above is written:

`00000 01111 11110 10101 01000 0000`

Problem 1

Write the following sentence in terms of **P**, **Q**, and **R**.

`10101 01010 11111 11101 01010 10000 00010 10101 01011 11111 10101 01010`

Answer to problem 1

This is just direct translation and the answer is **PQPRPQP**.

Problem 2

Here is a sentence which has been xored with a pseudorandom bitstream with approximately 10% 1's. Write the original sentence in terms of **P**, **Q**, and **R**.

`11011 11101 01010 10101 00010 10001 01010 00010 00010 10101 01011 11011`

Answer to Problem 2

This is the pseudorandom bitstream used in the problem:

`00100 00000 00000 00000 01000 00100 00000 10010 00000 00000 00000 00100`

and here's the original bitstream:

`11111 11101 01010 10101 01010 10101 01010 10000 00010 10101 01011 11111`

so that the bitstream is **QPPPRPQ**. One can guess this (and, really, this is *only* a guess!) by looking at the patterns. The first 7 bits in the bitstream of the problem statement are `11011 11` and since every bit there has a 90% chance of being correct, it is rather unlikely that this is a result of xoring with either **P** or **R**. For example, if we had started with **R**, we'd need to change from `00000 0` to `11011 1` and that would mean a total of 5 out of 6 "bitflips" – there's only one chance out of 100,000 (that's 10^5) of that happening. If we had started with **P**, we'd need to change from `10101 01` to `11011 11`. The number of bitflips here would be 4. The chance of that occurring is one chance out of 10,000. Compare that with starting with **Q**, where only one bitflip (one chance of 10) is necessary.

⋆⋆⋆⋆⋆

Lesson from problem 2

In real life, people look *very* carefully at the statistics of the pseudorandom bitstreams that are used. Even small systematic biases (1% more 1's than 0's, for example) can be easily exploited by cryptanalysis. Much more complex statistics than simple counting are explored. Correlations (relationships between the bits) are investigated quite closely.

Problem 3

The following bitstream is two sentences xored together. Write the original sentences in terms of **P**, **Q**, and **R** as well as you can.

`01010 10111 10101 01000 00000 00101 01000 00101 01001 01010 11110 10101`

Answer to problem 3

The two bitstreams which were created are
`10101 01010 11111 11101 01010 10000 00010 10101 01011 11111 10101 01010`
which is **PQPRPQP** and
`11111 11101 01010 10101 01010 10101 01010 10000 00010 10101 01011 11111`
which is **QPPPRPQ**.

The difference in length and pattern make it fairly easy to learn what the two bitstreams are. The solutions written above are *not* the only possibilities. The sequences **PQPPRQP** and **QPPRPPQ** give the same answer. There are also other valid answers. Cryptanalysis needs some inspired guessing or further information about either the meaning of the "language" or about the relative frequencies of **P** coming after **Q** versus **Q** coming after **P**: more statistical information. Such information is certainly recorded about any language of interest.

⋆⋆⋆⋆⋆

Lesson from problem 3

Many people think that a one-time pad constructed from a common text (such as a novel or a book of poems or a dictionary) is a good idea. This example should show that the statistics of a natural language are so rough (non-random) that natural language should *never* be used as the source of a random bitstream.

Problem 4

A pseudorandom bitstream with approximately 50% 1's and 50% 0's has been created. Two <u>different</u> sentences have been xored with it and the results are shown. Write the original sentences in terms of **P**, **Q**, and **R** .

`11110 00111 00101 10010 10010 10011 11101 11110 00100 10111 01110 11011 100`

`11110 00111 10000 10010 11000 00110 10111 10100 10101 11101 00100 01001 00`

Answer to problem 4

This is the pseudorandom bitstream used in the problem

`01011 01101 10000 11000 01101 01100 00010 10100 10000 10111 10001 00011 100` and following are the two original sentences (in order).

`10101 01010 10101 01010 11111 11111 11111 01010 10100 00000 11111 11000 000`

This sentence is **PPQQPRQR**.

`10101 01010 00000 01010 10101 01010 10101 00000 00101 01010 10101 01010 10`

This sentence is **PRPPRPP**.

The xor of the two sentences is

`00000 00000 10101 00000 01010 10101 01010 01010 10001 01010 01010 10010 10`

which must be the same as the xor of the two encrypted sentences. Why?

Digression on $\oplus$ (the xor symbol)

$\oplus$ is addition mod 2. So the entire definition of $\oplus$ is given by these equations:

$$0 \oplus 0 = 0; \quad 0 \oplus 1 = 1; \quad 1 \oplus 0 = 1; \quad 1 \oplus 1 = 0$$

$\oplus$ is used extensively in cryptography because $a \oplus b \oplus b = a$ for any a's and b's, so we can do the following:

$$\overbrace{a}^{\text{One original message bit}} \longrightarrow \overbrace{a \oplus b = m}^{\text{Encryption using } b} \longrightarrow \overbrace{m \ldots m}^{\text{Message transmission}} \longrightarrow \overbrace{m \oplus b = a \oplus b \oplus b = a}^{\text{Decryption using } b} \longrightarrow \overbrace{a}^{\text{Original message bit received.}}$$

End of digression on $\oplus$ (the xor symbol)

If a is a bit from the first sentence, A is a corresponding bit (in the same position) from the second sentence, and b is the corresponding bit from the bitstream which encrypts them both, then $a \oplus b$ and $A \oplus b$ are bits of the encrypted stream. If we xor these bits we get:

$$(a \oplus b) \oplus (A \oplus b) = a \oplus A \oplus b \oplus b = a \oplus A$$

so b has no influence on what's left. We can just use ideas about a and A to try to guess about them. In the case of the messages given, we are lucky (not very, given the statistics of this language!) that both of the messages begin with **P**. The next pattern in the xored bitstream is `10101` which suggests alternating agreement and disagreement among the bitstreams. Since `1` means disagreement of bitstreams, we examine our dictionary:

P=`1010101010` **Q**=`1111111` **R**=`000000`

and see that one of the sentences has the word **P** and one must have **R** because they disagree on the first bit of the third group. Now we continue, really guessing which of the sentences has **P** and which has **R** and following the consequences. Sometimes the wrong guess will be made so backtracking will need to be done: consideration of alternative possibilities.

⋆⋆⋆⋆⋆

Lesson from problem 4

Never use a pseudorandom bitstream more than once. Cryptanalysis can rapidly read both of the message streams – such a situation is sometimes called a "depth". There's almost no protection: a one-time pad should be used exactly once.