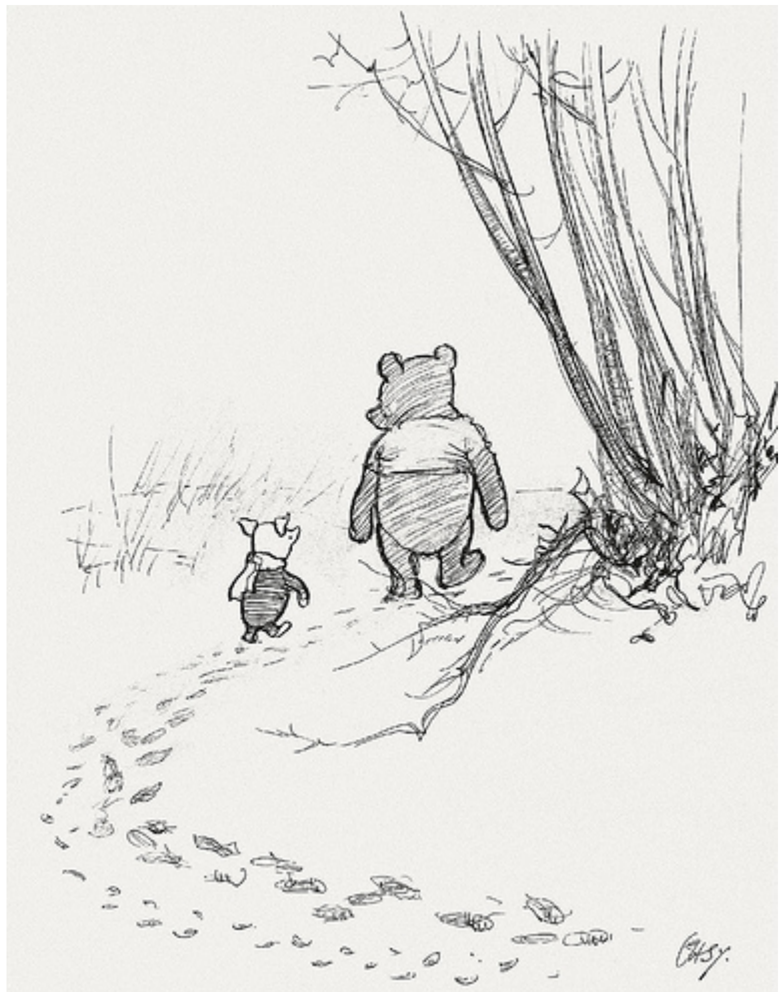
ECC



Error Detection

Bits in a word can be flipped.
It can happen because of noise on a bus or even the value of a bit being altered by the passage of a charged particle, from cosmic rays or from natural radiation.

Simplest solution is add an extra bit to a word. Set that bit so that there are either an even number of odd number of bits in the extended word.

1	**1**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**

I have chosen to make an even parity word.
If any bit is altered including the parity bit, we can tell that the word is incorrect.

BUT we can't tell which bit is wrong.

It two bits are flipped the parity will be correct again.

CANNOT tell if two bits are flipped

Error Detection

The greater the number of bits which are covered by a parity bit, the more efficient the code is, but the more likely it is to miss a two bit error.

1 bit for 8 bits – 88.9% of bits are data
1 bit for 16 bits – 94.1% of bits are data
1 bit for 32 bits – 96.9% of bits are data

But of course a string of 32 bits is 4 times as likely to have an error as 8 bits.

Balance error rate against storage efficiency

Use 3 bits per data bit (repetition)

$\boxed{0\ 0\ 0}$ $\boxed{1\ 1\ 1}$

Take a majority of the bits as the answer – single bit errors can be corrected, but at the expense of reduced information transfer.

4 ECC

Richard Hamming
1950

Error correcting codes.

It is more useful if you can identify an error and correct it.

The simplest ECC is called a Hamming Code.

I will describe how to construct a Hamming code and take as an example a Hamming (15,11) code.
This has 4 parity bits and 11 distinct numbers – if we use bit zero as an overall parity bit, it can detect up to 2 errors and correct a single error.

Its efficiency is better than a (3,1) repetition 68% as opposed to 33%, but the chances of an error are about 5 times as great.

Building the code

Take a 16 (2^n) bit word.

Label the bits in binary

0000, 0001, 0010, 0011, 0100, 0101,

0110, 0111, 1000, 1001, 1010, 1011,

100, 1101, 1110, 1111

Each word with only 1 bit set is a parity bit. The rest are data.

0000 is an overall parity bit – ignore at present.

4 (n parity bits)
11 data bits ($2^n - n - 1$)

Efficiency = $(2^n - n - 1)/(2^n)$

Called a (15,11) code

Building the code

Hamming codes have a minimum distance of three – at least three bits have to be flipped to get from one legal state to another.

If you are one away from a state you can get back to it. Correct an error

0000, 0001, 0010, 0011, 0100, 0101,

0110, 0111, 1000, 1001, 1010, 1011,

1100, 1101, 1110, 1111

Consider the 1000 parity bit.
That acts as a parity bit for all the bits with 1 in the 4th position of the binary expansion.

9, 10, 11, 12, 13, 14, 15

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
							0	0	1	1	0	0	1	1

Building the code

0000, 0001, 0010, 0011, 0100, 0101,

0110, 0111, 1000, 1001, 1010, 1011,

1100, 1101, 1110, 1111

Consider the 0100 parity bit.

5, 6, 7, 12, 13, 14, 15

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
				0	1	1	1				0	0	1	0

Consider the 0010 parity bit.

3, 6, 7, 10,11,14,15

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	0			1	1			0	0			1	0

Building the code

0000, 0001, 0010, 0011, 0100, 0101,

0110, 0111, 1000, 1001, 1010, 1011,

1100, 1101, 1110, 1111

Finally 0001 parity bit.

3, 5, 7, 9, 11, 13, 15

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0		**0**		**0**		**1**		**1**		**0**		**0**		**0**

Building the code

The 16th bit is the parity for the whole word

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	1	1	1	0	0	0	0	1	1	1	1	0	1	1	0

Using the code

The parity bit for all the sets of data bits is calculated and compared with the parity bit.

Building a word

Want to send the year 2016.
Place it in the data words.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
		0		1	1	1		1	1	1	0	0	0	0	

Now calculate the parity bit

Note every bit is covered by more than 1 parity bit.

P1	3, 5, 7, 9, 11, 13, 15	0
P2	3, 6, 7, 10, 11, 14, 15	0
P4	5, 6, 7, 12, 13, 14, 15	1
P8	9, 10, 11, 12, 13, 14, 15	1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	

Add word parity bit

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0

Correcting a single bit error

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0

P1	3,	5,	7,	9,	11,	13,	15	0
P2	3,	6,	7,	10,	11,	14,	15	0
P4	5,	6,	7,	12,	13,	14,	15	1
P8	9,	10,	11,	12,	13,	14,	15	1

Any single flip will be indicated by bit 16

Bit 16 flipped – Pn correct.   Correct bit16

If a data bit is flipped then it affects at least 2 parity bits
Flip a parity bit – then Pn is wrong and bit 16 is also wrong

Correct the parity bit

Flip bit 9. P1 and P4 and 16 are wrong.

The parity bits are the binary representation of the flipped bit   1001

Detecting a two bit error

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0

P1	3, 5, 7, 9, 11, 13, 15	0
P2	3, 6, 7, 10, 11, 14, 15	0
P4	5, 6, 7, 12, 13, 14, 15	1
P8	9, 10, 11, 12, 13, 14, 15	1

Two data words flipped some subset of Pn will be wrong, but P16 is correct.

Bit 16 is acting as an error counter.

If it is wrong and something else is wrong then it is a single bit error.
If it is right and something else is wrong, then there are two errors

Forward Error Correction

Data can be stored or transferred with an error correcting code

It is possible to detect a certain number of errors in such data and it is possible to correct a smaller number of errors.

In general it will be the controller unit which is responsible for correcting the errors, or signaling that there are too many errors to correct.

HDD controller
SDD controller
Memory controller
Network card.

For any of them when they detect an error, they correct the error before forwarding it to the CPU

Hamming codes are often used for solid state drives.

Different techniques

The correction technique chosen is a balance between efficiency and reliability and the penalty for a mistake

Techniques should also be matched to expected error types

Network signals are often subject to a burst of noise.

Errors are not independent, they are strongly correlated.

There exist coding schemes which distribute the data over a large number of words, errors may then be recovered from, even if several dozen bits in a row are corrupt.

Such codes are beyond this course

DEC tapes

Like a slow disk drive

19mm 184k

6.2mm - hole not a problem