Hamming Code and Error Correction

Sponsored in part by:

YouTube

Outline

A little background error correction (in computers)

The early days of error correction

Different methods of error correction

Hamming Code

Conclusion

A little background

- What's a communication error?- Choppy phone connection "What did you just say?"
- Error Correction (ex] repeating a word)
- Used in all kinds of computing/ communication (telecommunication, computer ram)
- (Example: Scratched CD still playing properly)

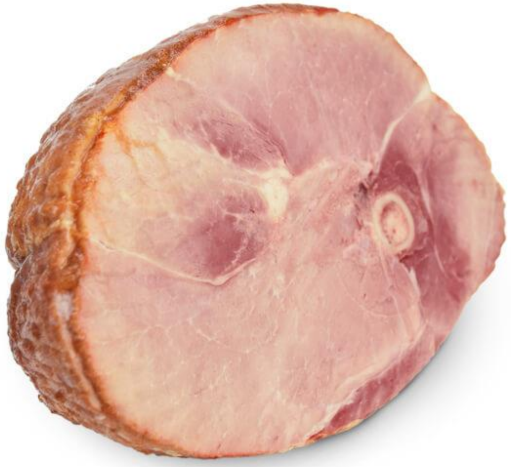
Methods of Correction/ snags

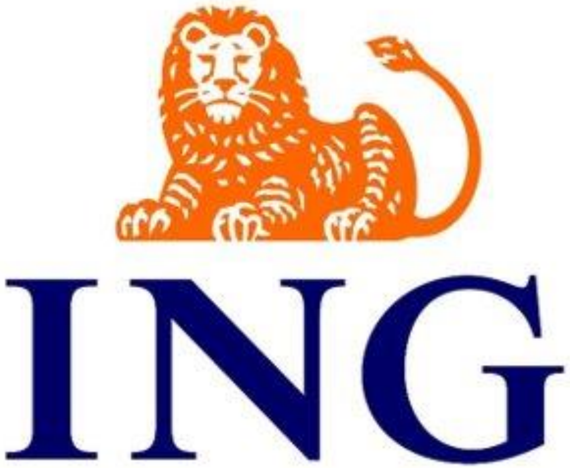
- Repetition Code (redundant, which version is right?, inefficient for many applications)
- Parity Bits (only tells us if number of 1s are odd or even)
- Other methods include: Cyclic Redundancy Codes, Checksums, Cryptographic Hash Functions.(developed more recently. 1970+)

World's first coding errors

- Richard Hamming-Bell Labs
- Punchcards no-hole vs hole 0 or 1
- Holes missed/accidentally punched
- Whole system stalls until error is found
- Hamming sets out on mission to find solution

Richard

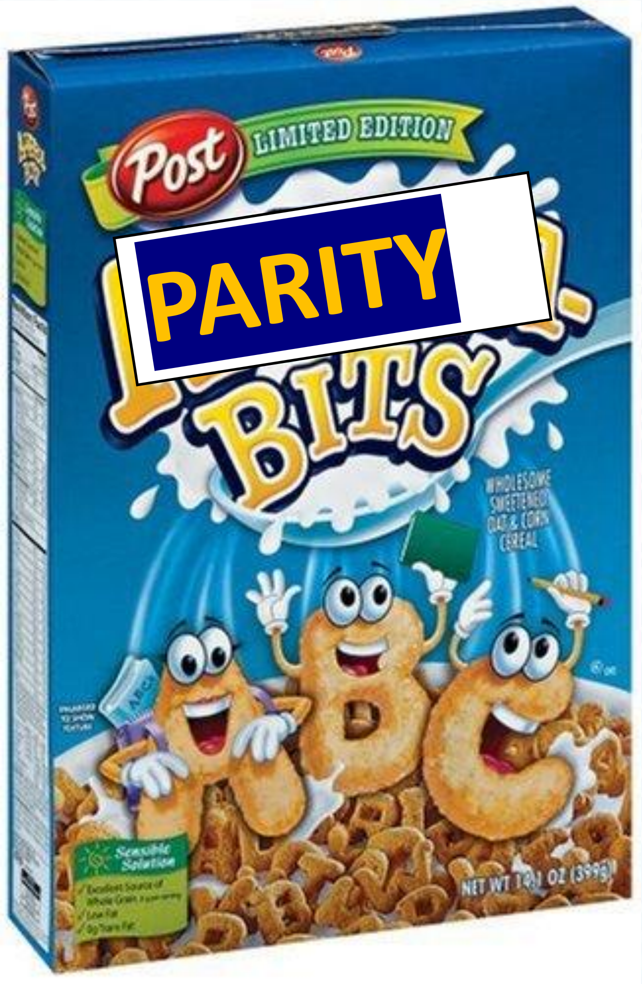




At

Discovered a Method For

USING


Post
LIMITED EDITION
PARITY
BITS
WHOLESOME SWEETENED OAT & CORN CEREAL
Sensible Solution
NET WT 14.1 OZ (399g)

Hamming Code

- Named after Richard Hamming
- Parity Bits: Basic Premise for Hamming Code
- Useful in situations with low error rate (such as in computer RAM)
- Note: Hamming code will only correct 1-bit errors
- We will go over the most common examples of Hamming code, the Hamming(7,4) implementation; four bits and three parity bits.

What’s a parity bit?

par·i·ty

[ˈperədē]

NOUN

mathematics

(of a number) the fact of being even or odd.

Parity tells us whether the number of 1s in a string is odd or is even. Parity bit of 0 means even number of ones, 1 means odd number of 1s.

0 = Even number of 1s.

1 = Odd number of 1s

Hamming Code: How It Works

1. Given a sequence of 0s and 1s, number the position of each bit and mark all 2^n positioned bits as a parity bit.

Ex 1)

0	1	0	0	1	0	1
1	2	3	4	5	6	7
1	2	3	4	5	6	7

Ex 2)

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

Hamming Code: How It Works

For each 2^n parity bit, starting at bit 2^n record 2^n bits and skip 2^n bits.

Ex) 2^0 bit

0	1	0	0	1	0	1
1	2	3	4	5	6	7
1	2	3	4	5	6	7
0		0		1		1

2^1 bit

0	1	0	0	1	0	1
1	2	3	4	5	6	7
1	2	3	4	5	6	7
	1	0			0	1

Result for 2^0: 0011

Result for 2^1: 1001

Hamming Code: How It Works

Result for 2^2

0	1	0	0	1	0	1
1	2	3	4	5	6	7
1	2	3	4	5	6	7
			0	1	0	1

Result for 2^0: 0011- Parity bit is zero and we have 2 (even) 1s.

Result for 2^1: 1001- One as parity bit and we have a single 1 (odd) 1.

Result for 2^2: 0101- Parity bit is zero and we have 2 (even) 1s.

Since all the parity bits check out, there aren't any errors.

Time for an example on the board...

And that's how error correction works with Hamming Code.

Thanks for listening! ☺

Sources

- http://www.ams.org/publicoutreach/feature-column/fcarc-errors6
- https://www.geeksforgeeks.org/computer-network-hamming-code/
- http://math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf (A Mathematical Theory of Communication)

The illustrated version:

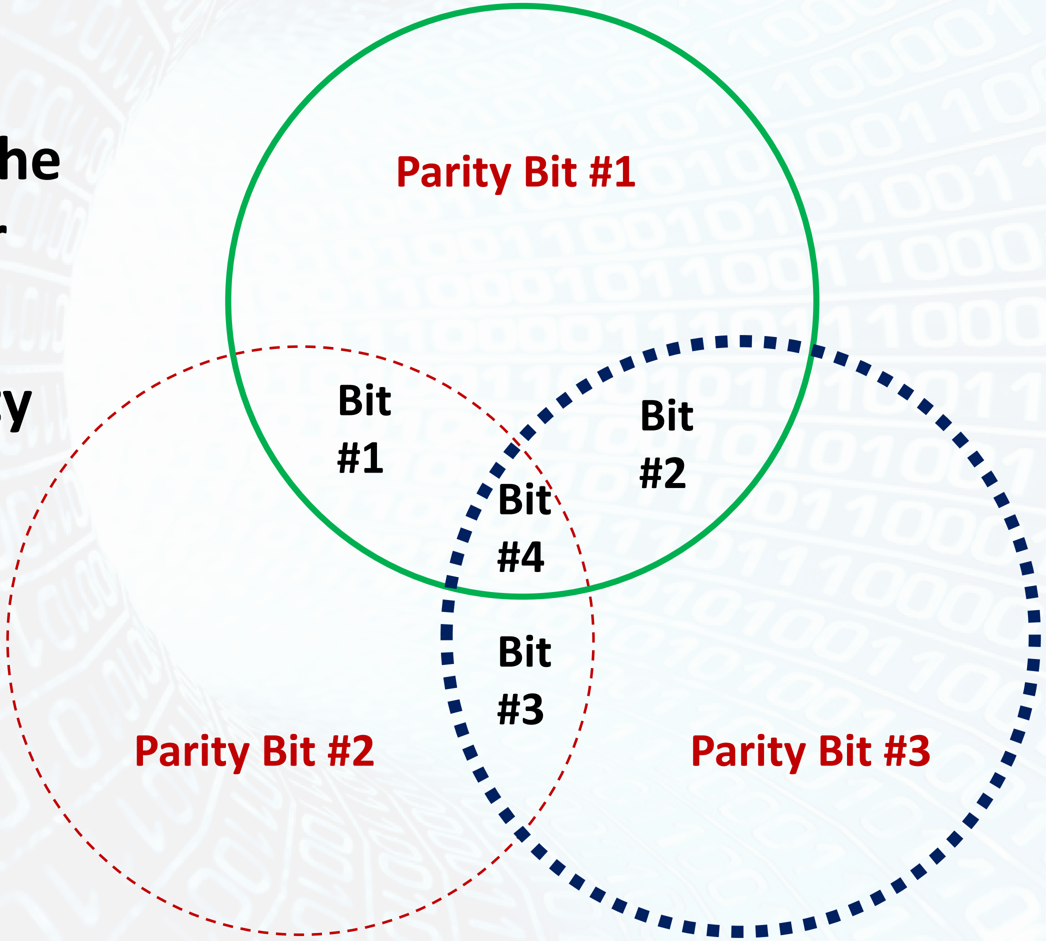
- Before we describe the algebra of Hamming codes, we first visualize the calculation of the parity bits using Venn diagrams. As an example, suppose we wish to send the 4-bit message `1101`. We associate each of the four message bits with a specific intersection region of three pairwise overlapping circles, reading bits from left to right:

- The Hamming code adds three parity bits so that each of the three circles has *even parity*.
- That is, the sum of the four bits contained in each of the three circles is even:
- For this example, the three parity bits are 1 (top), 0 (left), and 0 (right). So, to send a version of the message 1101 that is robust against single-bit errors, the actual message we send is the 7-bit message 1101100.
- Now, imagine this picture is transmitted over a noisy communication channel, and that one bit is corrupted so that the following picture arrives at the receiving station (corresponding to 1001100):

To visualize: Consider 4 bits of data we want to convert to Hamming Code

Place the bits in the following manner

Then add the parity bits


Parity Bit #1
Bit #1
Bit #2
Bit #4
Bit #3
Parity Bit #2
Parity Bit #3