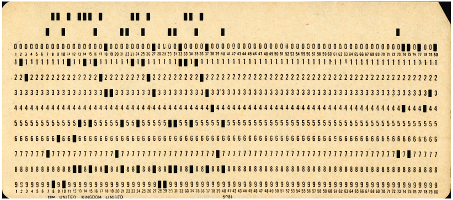
Inrtoduwtion to Erorr Dwtetcion and Erorr Czrrectmon

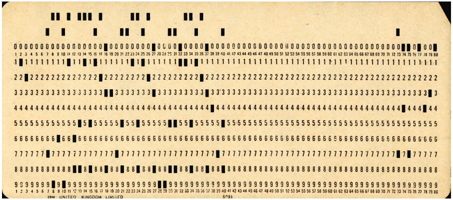
Setevn .J Mzlwer
sjm1@williams.edu
http://www.williams.edu/Mathematics/sjmiller



MothCulb, Univrseity of Mihciagn, Otcober 1, 2012

Introduction to Error Detection and Error Correction

Steven J. Miller
sjm1@williams.edu
http://www.williams.edu/Mathematics/sjmiller



Mathclub, University of Michigan, October 1, 2021

Introduction

Cryptography Basics

Enough to send 0's and 1's:

◇ A = 00000, B = 00001, C = 00010, . . .
Z = 11010, 0 = 11011, 1 = 11100, ...

Two major issues:

◇ Transmit message so only desired recipient can read.

◇ Ensure correct message received.

Cryptography Basics

Enough to send 0's and 1's:

- A = 00000, B = 00001, C = 00010, ...
  Z = 11010, 0 = 11011, 1 = 11100, ...

Two major issues:

- Transmit message so only desired recipient can read.
- Ensure correct message received.

Bit Error Dangers: RSA

If receive wrong bit in RSA, message completely different.

Bit Error Dangers: RSA

If receive wrong bit in RSA, message completely different.

Secret: $p = 15217$, $q = 17569$, $d = 80998505$.
Public: $N = pq = 267347473$, $e = 3141593$.
Note: $ed = 1 \bmod (p-1)(q-1)$.
Message: $M = 195632041$, send $M^e \bmod N$ or
$X = 121209473$.
Decrypt: $X^d \bmod N$ or 195632041.

Bit Error Dangers: RSA

If receive wrong bit in RSA, message completely different.

Secret: $p = 15217$, $q = 17569$, $d = 80998505$.
Public: $N = pq = 267347473$, $e = 3141593$.
Note: $ed = 1 \mod (p-1)(q-1)$.
Message: $M = 195632041$, send $M^e \mod N$ or
$X = 121209473$.
Decrypt: $X^d \mod N$ or 195632041.

Imagine receive $\widetilde{X} = 121209483$.
Message 195632041
Decrypts 121141028, only two digits are the same!

Outline

Will concentrate on Error Detection and Correction.

- Detection: Check Digit
- Correction: Majority Rules and Generalization

Check Digit

Check Digit

If easy to read again, just need to detect error.

Check Digit

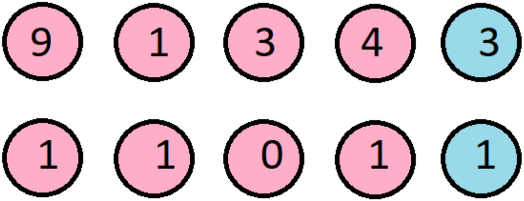
If easy to read again, just need to detect error.

Think scanner at a supermarket....

Check Digit

If easy to read again, just need to detect error.

Think scanner at a supermarket....

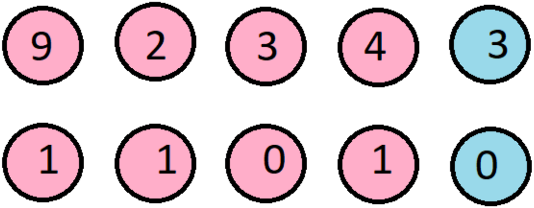


Last digit makes sum 0 mod 10 (or 0 mod 2).

Check Digit

If easy to read again, just need to detect error.

Think scanner at a supermarket....



Last digit makes sum 0 mod 10 (or 0 mod 2).

Next Steps

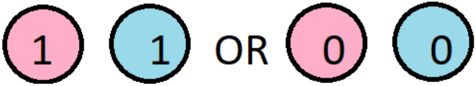
More involved methods detecting more: The Verhoeff algorithm catches single digit errors and flipping adjacent digits: https://en.wikipedia.org/wiki/Verhoeff_algorithm.

Want to detect where the error is:

Next Steps

More involved methods detecting more: The Verhoeff algorithm catches single digit errors and flipping adjacent digits: https://en.wikipedia.org/wiki/Verhoeff_algorithm.

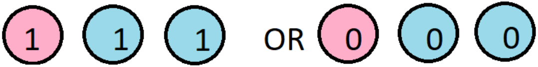
Want to detect where the error is: Tell me twice!



Majority Rules

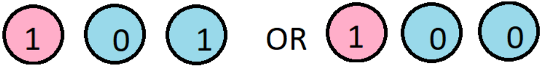
Tell Me Three Times

Tell Me Three Times detects and *probably* corrects (need probability of an error small).



Tell Me Three Times

Tell Me Three Times detects and *probably* corrects (need probability of an error small).



Tell Me Three Times

Crucially uses binary outcome: https://www.youtube.com/watch?v=RerJWv5vwxc and https://www.youtube.com/watch?v=vWCGs27_xPI.

What is the problem with this method?

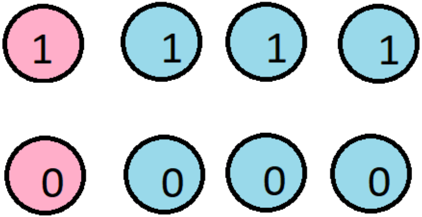
Tell Me Three Times

Crucially uses binary outcome: https://www.youtube.com/watch?v=RerJWv5vwxc and https://www.youtube.com/watch?v=vWCGs27_xPI.

What is the problem with this method? Only one-third is information.

How can we do better?

Tell Me *n* Times

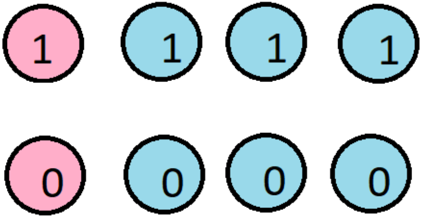


Tell Me Four Times: only 25% of message is data (general case just $1/n$).

Want to correct errors but still send a lot of information.

What's a success?

Tell Me n Times



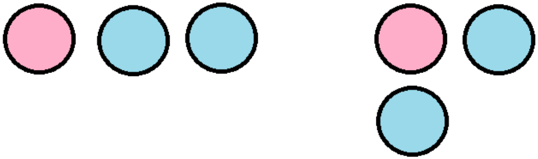
Tell Me Four Times: only 25% of message is data (general case just $1/n$).

Want to correct errors but still send a lot of information.

What's a success? Greater than 50% is data.

Tell Me Three Times (revisited)

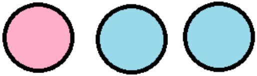
Let's revisit Tell Me Three Times:

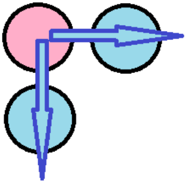


How should we do two data points?
How many check digits do you expect?

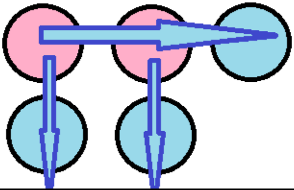
Tell Me Three Times (revisited)

Let's revisit Tell Me Three Times:



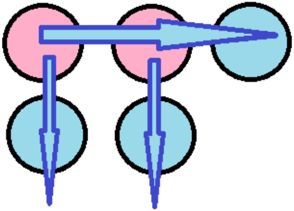


How should we do two data points?
How many check digits do you expect?



Two of Five

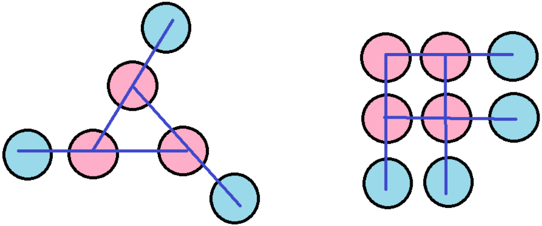
This is better: 2 of 5 or 40% of message is data!



Unfortunately still below 50%.

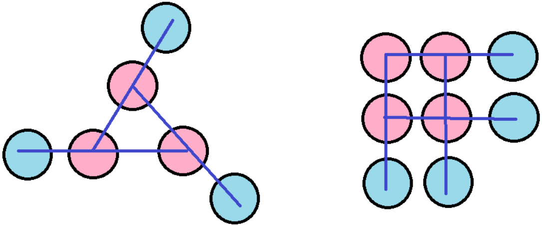
How many data points should we try next: 3, 4, 5, ...?

Three and Four Bits of Data



Which is better?

Three and Four Bits of Data

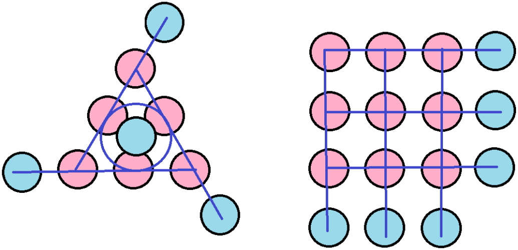


Which is better? Both 50% but fewer needed with triangle.

What should we do next: 5, 6, 7, 8, 9, ...?

Triangle and Square Numbers

$T_n = n(n+1)/2$ and $S_n = n^2$.

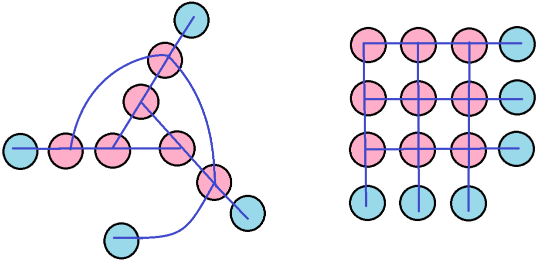


Both give 60% of the message is data. Can we continue?

Data on exactly two lines, check bits on one.

Triangle and Square Numbers

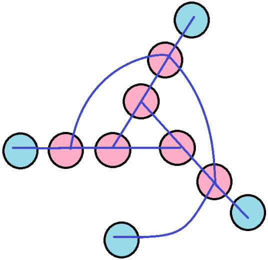
$T_n = n(n+1)/2$ and $S_n = n^2$.



Both give 60% of the message is data. Can we continue?

Data on exactly two lines, check bits on one.

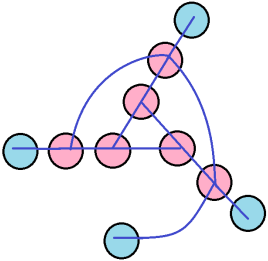
Triangle and Square Systems

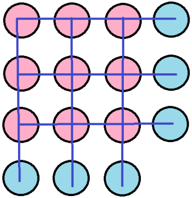


Triangle: $T_n = n(n+1)/2$ data, $n+1$ check, so $(n+2)(n+1)/2$ bits total and $n/(n+2)$ information.

Square: $S_n = n^2$ data, $2n$ check, so $n^2 + 2n$ bits total and $n/(n+2)$ information.

Triangle and Square Systems



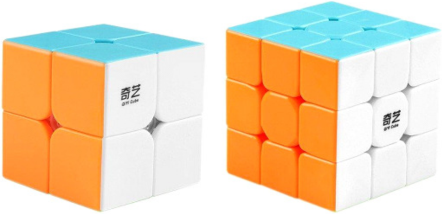


Can get as high a percentage information as desire, at a cost of longer string (and thus more likely to have two errors).

Generalizations

What is a better geometry to use?

Generalizations

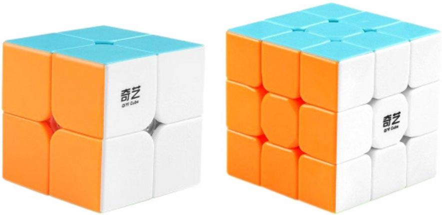


$2 \times 2 \times 2$: 8 data points, 6 check bits (for planes): info is $8/14 \approx 57\%$.

$3 \times 3 \times 3$: 27 data points, 9 check bits (for planes): info is $27/36 = 75\%$.

For 6×6 data square info is $36/48 = 75\%$, for T_7 is $28/36 \approx 77.78\%$.

Generalizations

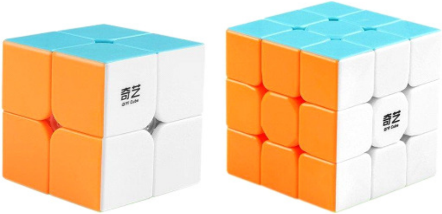


$4 \times 4 \times 4$: 64 data points, 12 check bits: info is $64/76 \approx 84.21\%$.

For 9×9 data square info is $81/99 \approx 81.82\%$.

For T_{11} triangle: 66 data points, info is $66/79 \approx 83.54\%$.

Generalizations

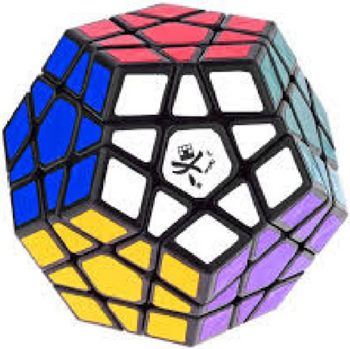


$n \times n \times n$: n^3 data points, $3n$ check bits: info is $n^2/(n^2+3)$.

Better percentage is information for large n; how should we generalize?

Generalizations

What is a better geometry to use?

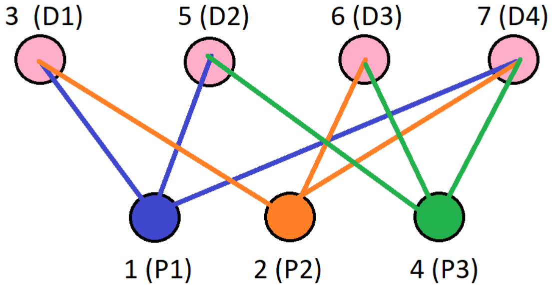


Other Approaches

Hamming Codes: Can send a message with 7 bits, 4 are data, and can correct one error: https://en.wikipedia.org/wiki/Hamming_code.

Extended binary Golay code: Can send a message with 24 bits, 12 are data, can correct any 3-bit errors and can detect some other errors: https://en.wikipedia.org/wiki/Binary_Golay_code.

Manhamming



- If no errors, all correct.
- If only one color error, is P1, P2 or P3.
- If just blue and orange is D1.
- If just blue and green is D2.
- If just orange and green is D3
- If all wrong is D4.

Comparison

Say want to transmit around $2^{12} = 4096$ bits of data.

Can do a square and cube; the Hamming code will do $2^{12} - 1 - 12$.

- Square: 4096 out of 4224 data: 96.9697%.
- Cube: 4096 out of 4144 data: 98.8417%.
- Hamming: 4083 out of 4095 data: 99.707%.

All converge to 100%, difference narrows as size increases.

Interleaving

Say transmit

01111010010101010101010101010101010101011110...

but a localized burst of noise, receive

01110111010101010101010101010101010101011110...

Interleaving

Transmit every fourth:

- 01000000001 $\mapsto$ 00000000001
- 10111111111 $\mapsto$ 11111111111
- 11000000001 $\mapsto$ 11000000001
- 10111111110 $\mapsto$ 11111111110

Steganography

Can you see the cat in the tree?

Transmitting Images

How to transmit an image?

- Have an $L \times W$ grid with LW pixels.
- Each pixel a triple, maybe (Red, Green, Blue).
- Often each value in $\{0, 1, 2, 3, \ldots, 2^n - 1\}$.
- $n = 8$ gives 256 choices for each, or 16,777,216 possibilities.

Steganography

Steganography: Concealing a message in another message: https://en.wikipedia.org/wiki/Steganography.

Steganography

Steganography: Concealing a message in another message: https://en.wikipedia.org/wiki/Steganography.

Take one of the colors, say red, a number from 0 to 255.

Write in binary: $r_7 2^7 + r_6 2^6 + \cdots + r_1 2 + r_0$.

If change just the last or last two digits, very minor change to image.

Can hide an image in another.

If just do last, can hide a black and white image easily....

Can you see the cat in the tree?

Can you see the cat in the tree?