Ergo…

- Error detection è utile quando il canale ha ***pochi errori*** (è molto affidabile), oppure quando un errore non è ***critico***
- Ma spesso serve anche fare **error correction**, cioè cercare anche di correggere gli errori

Error correction…

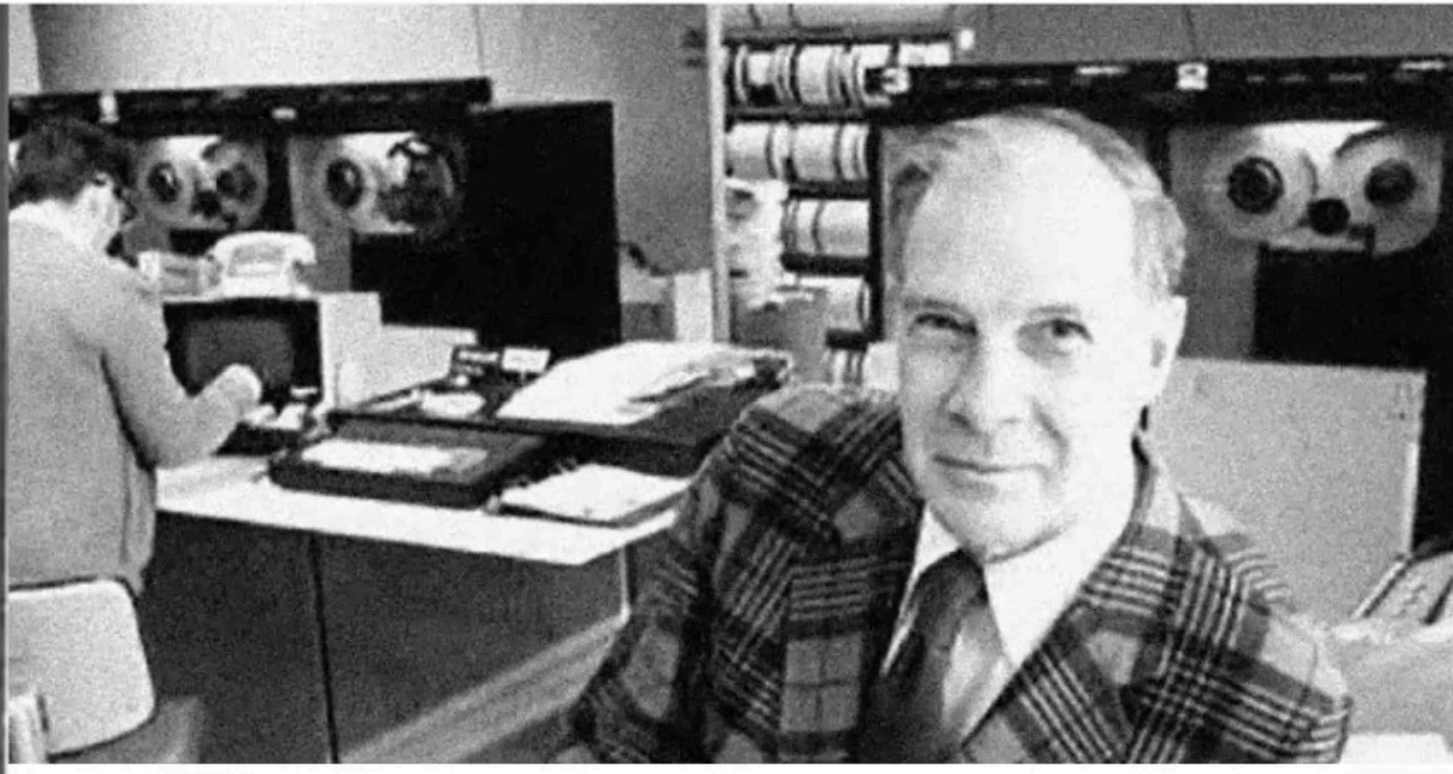
- Anche perché, nelle reti, quando invece il canale non è molto affidabile, ritrasmettere troppi pacchetti diventerebbe ***oneroso***: facendo direttamente ***correzione degli errori*** sui pacchetti, si evita la ritrasmissione

Esempio

- Una semplice tecnica è quella del bit di parità (**parity bit**):
- Ogni tot bits (diciamo, **m**), ***inseriamo*** uno 0 o un 1 a seconda che la somma degli m bits precedenti sia ***pari*** o ***dispari***
- m=2 : 01 → 011 10 → 101

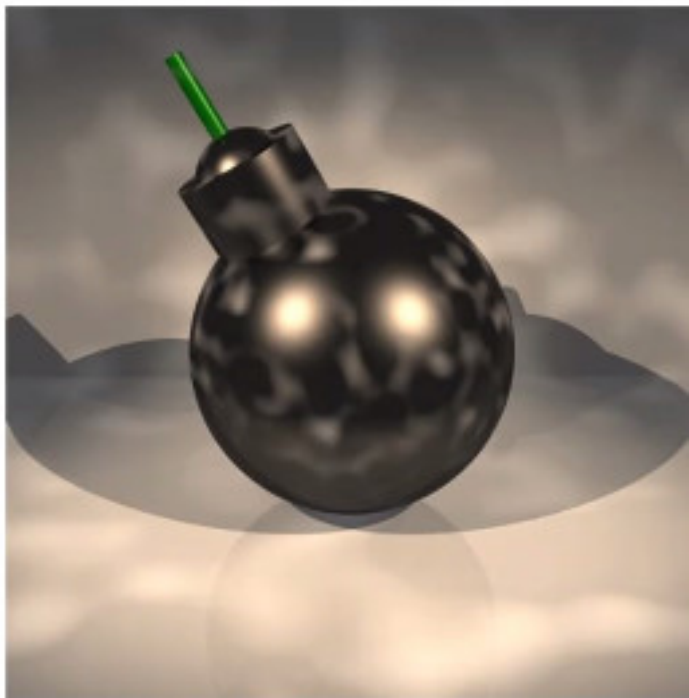
E allora?
Come ne troviamo di migliori?

Hamming…



Error control: distanze

- Prima di procedere, ci occorre una ***misura del "danno"*** che un errore può fare
- Gli errori ***meno gravi*** sono quelli che danneggiano ***un*** solo bit del messaggio…
- … poi seguono quelli che ne danneggiano ***due***, etc etc



La distanza

- Diciamo allora che due messaggi (di lunghezza uguale ovviamente) sono ***distanti 1*** se sono diversi solo per 1 bit, …
- ***Distanti 2*** se sono diversi per 2 bits, etc etc
- Questa misura di distanza si chiama **distanza di Hamming**

Le tecniche di error control…

- Sono quindi ***più o meno potenti*** a seconda di quanta distanza di Hamming riescono a sopportare nei messaggi (cioè, di quant'è la gravità massima dell'errore che riescono a sopportare)

Error detection

- Vediamo allora cosa si può fare nel caso dell'error detection
- Ovviamente, dovremo avere ***dell'informazione extra*** rispetto ai dati (come nel caso del framing) che ci permetta stavolta di trovare gli errori

Error detection

- Avremo quindi un ***encoding*** (in cui aggiungiamo la protezione dagli errori), e poi una fase di ***decoding*** dove ci liberiamo dalla protezione dagli errori e teniamo il dato originale

Error detection

- L'error detection sarà quindi nella fase di decoding, dove il ricevente controllerà se ci sono stati errori
- Modo semplice? Basta ad esempio verificare se il messaggio arrivato non può essere stato creato con l'encoding (messaggio "illegale")

Ma...

- Questa è una condizione ***sufficiente***, non ***necessaria***
- Potrebbero esserci talmente tanti errori da trasformare un messaggio "legale" in un altro messaggio "legale"
- Quindi, quand'è che siamo sicuri di trovare l'errore?
- Dobbiamo limitare la potenza dell'errore!

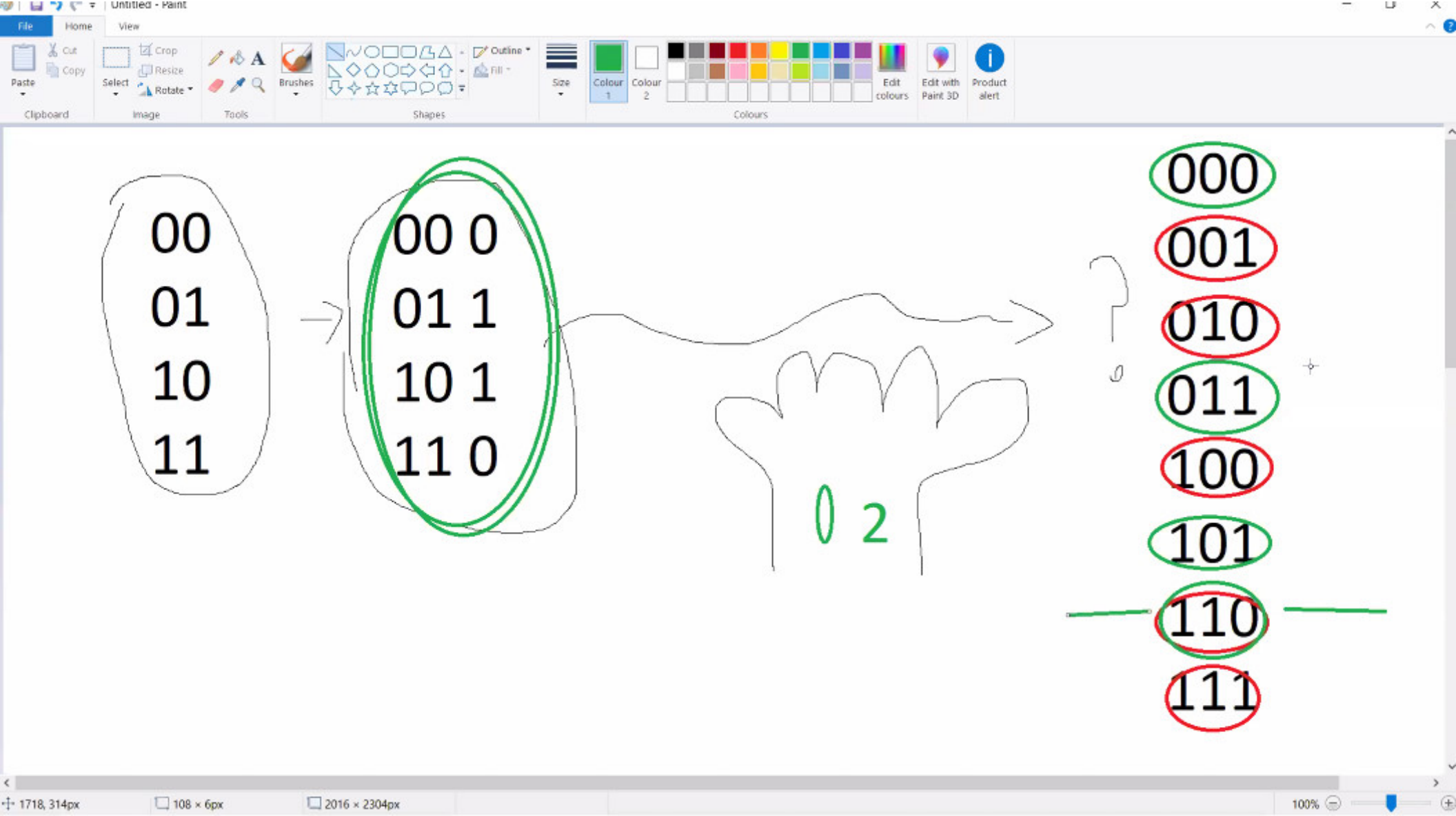
Riformulando...

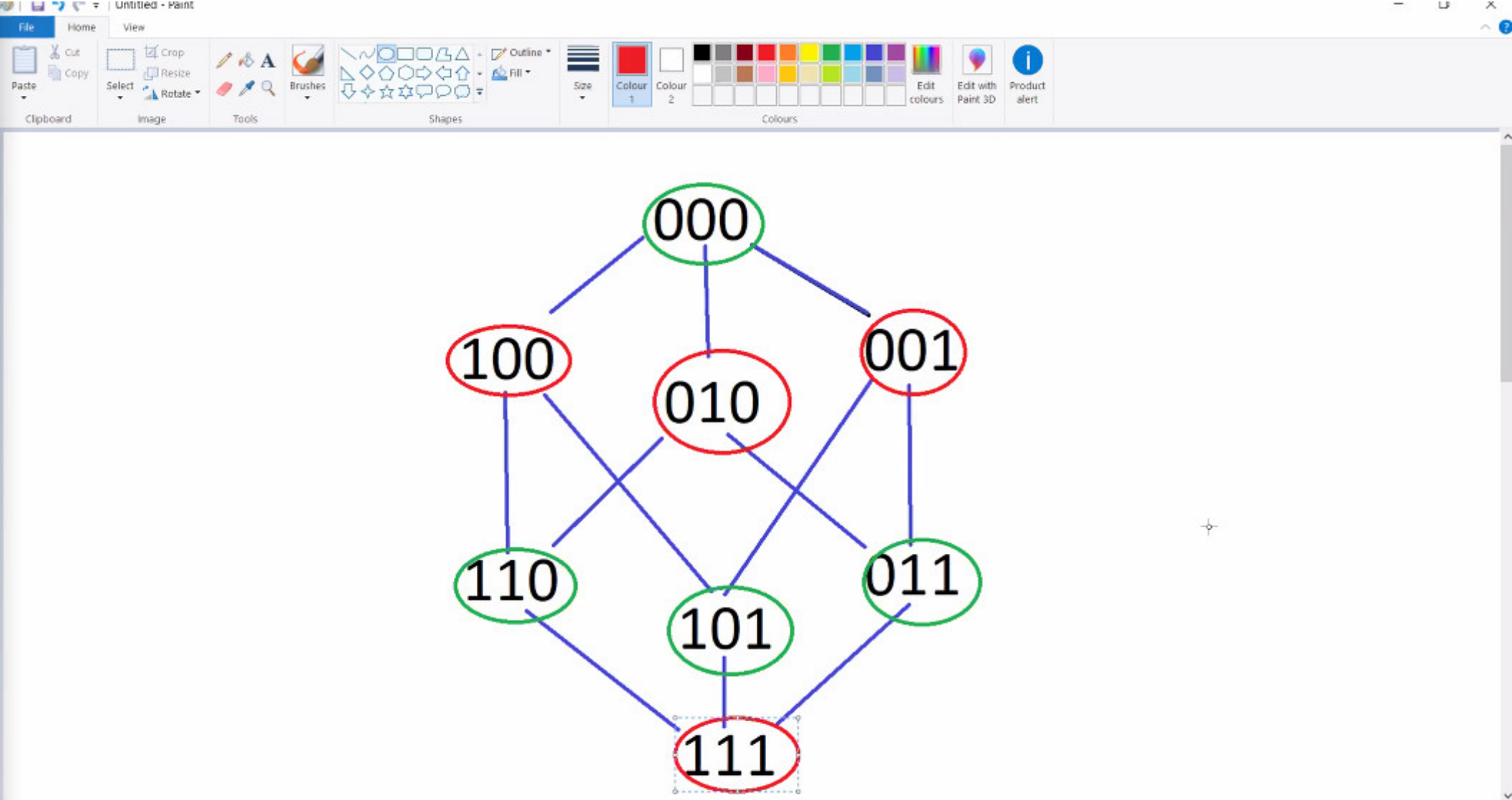
- Dato un certo modo di proteggere dagli errori (encoding/decoding) vorremmo sapere: qual'è il massimo errore che riusciamo a trovare?
- Vediamo con un esempio…

Rivediamo…

- La tecnica del **parity bit**
- Ad esempio il parity bit 2 (ogni 2 bit inseriamo il bit di parità)
  m=2 : 01 → 011 10 → 101

Disegnamolo...


Untitled - Paint
File
Home
View
Paste
Cut
Copy
Select
Crop
Resize
Rotate
Brushes
Outline
Fill
Size
Colour 1
Colour 2
Edit colours
Edit with Paint 3D
Product alert
Clipboard
Image
Tools
Shapes
Colours
00
01
10
11
00 0
01 1
10 1
11 0
0 2
000
001
010
011
100
101
110
111
1718, 314px
108 × 6px
2016 × 2304px
100%


Untitled - Paint
File
Home
View
Cut
Copy
Paste
Clipboard
Crop
Resize
Rotate
Select
Image
Brushes
Tools
Outline
Fill
Shapes
Size
Colour 1
Colour 2
Edit colours
Edit with Paint 3D
Product alert
Colours
000
100
001
010
110
011
101
111
1525, 499px
157 × 94px
2016 × 2304px
100%

Quanto è potente il parity bit "m"?

- Facile vedere che, **qualunque sia l'm**:
- Presi due messaggi diversi, ***i loro messaggi codificati sono a minimo a distanza 2***
- → anche se c'è un errore (1 bit), un messaggio codificato non può mai diventare un altro messaggio!