An introduction to error-correcting codes and some current day applications

dr. Relinde Jurrius

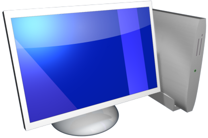
University of Neuchâtel, Switzerland

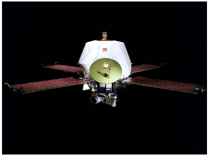
December 20, 2016

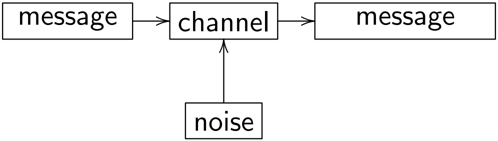
S W I T E E L R A N D

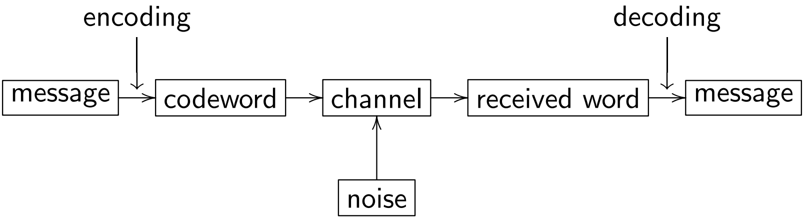
S W I T E E L R A N D

Redundancy





message
channel
message
noise


encoding
decoding
message
codeword
channel
received word
message
noise

$$0 \longrightarrow 00000$$

$$1 \longrightarrow 11111$$

$$0 \longrightarrow 00000$$

$$1 \longrightarrow 11111$$

00000 ?

01100 ?

10111 ?

0	$\longrightarrow$	00000		00000 ?	$\longrightarrow$	0
1	$\longrightarrow$	11111		01100 ?		
				10111 ?		

$$0 \longrightarrow 00000 \qquad\qquad 00000\ ? \longrightarrow 0$$

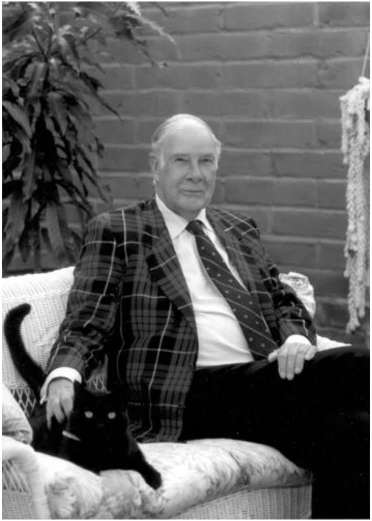
$$1 \longrightarrow 11111 \qquad\qquad 01100\ ? \longrightarrow 0$$

$$10111\ ?$$

0	$\longrightarrow$	00000		00000 ?	$\longrightarrow$	0
1	$\longrightarrow$	11111		01100 ?	$\longrightarrow$	0
				10111 ?	$\longrightarrow$	1

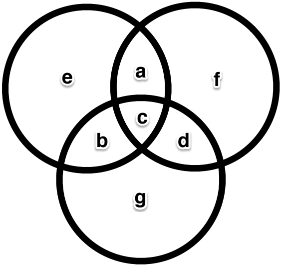
$$
\begin{array}{lll}
0 & \longrightarrow & 00000 \\
1 & \longrightarrow & 11111
\end{array}
\qquad
\begin{array}{lll}
00000\ ? & \longrightarrow & 0 \\
01100\ ? & \longrightarrow & 0 \\
10111\ ? & \longrightarrow & 1
\end{array}
$$

Redundancy: $\frac{4}{5}$

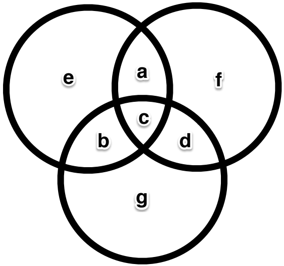


Richard Hamming
(1915–1998)

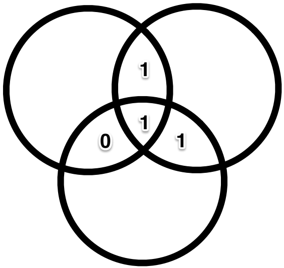
Bell Labs, ca. 1950

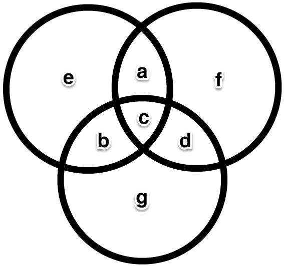


a	b	c	d	e	f	g
1	0	1	1			

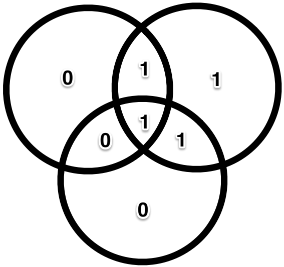


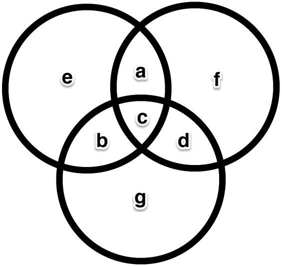
a	b	c	d	e	f	g
1	0	1	1			



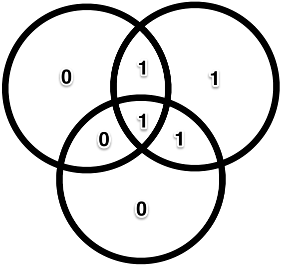


a	b	c	d	e	f	g
1	0	1	1			



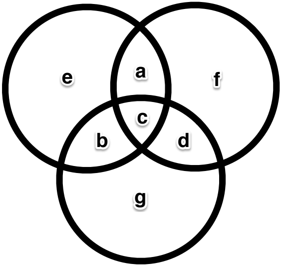


a	b	c	d	e	f	g
1	0	1	1			

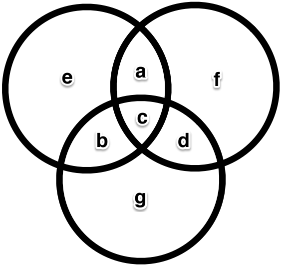


$$1011 \longrightarrow 1011010$$

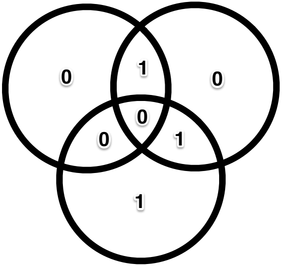
Redundancy: $\dfrac{3}{7}$

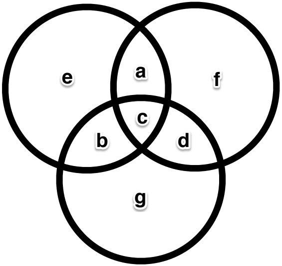


a	b	c	d	e	f	g
1	0	0	1	0	0	1

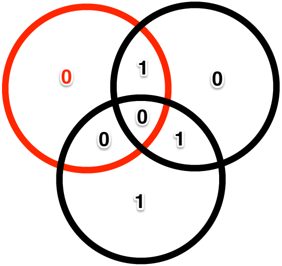


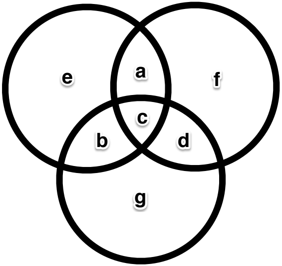
a	b	c	d	e	f	g
1	0	0	1	0	0	1



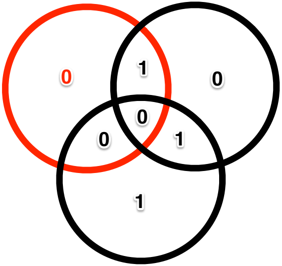


a	b	c	d	e	f	g
1	0	0	1	0	0	1





a	b	c	d	e	f	g
1	0	0	1	0	0	1



$1001001 \longrightarrow 1001{\color{red}1}01$

$1001101 \longrightarrow 1001$

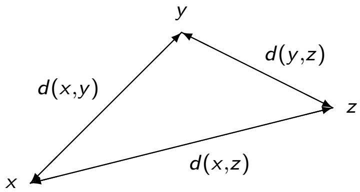
Low redundancy

Large differences between codewords

Fast encoding / decoding

Distance function $d(x, y)$ is a *metric* if:

$$d(x, y) \geq 0 \text{ with equality iff } x = y$$
$$d(x, y) = d(y, x)$$
$$d(x, y) + d(y, z) \geq d(x, z)$$



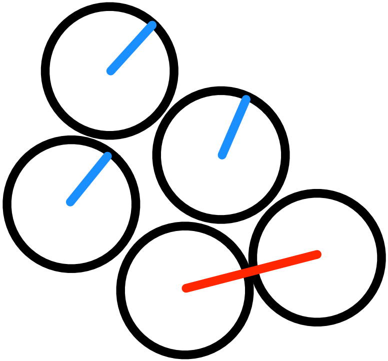
Alphabet $\mathcal{Q}$

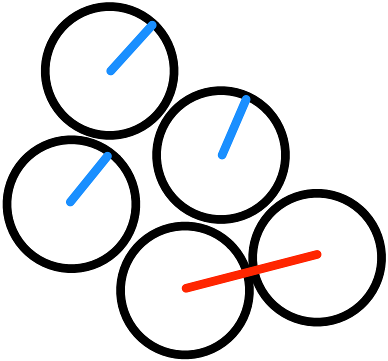
Length n

Hamming metric on $\mathcal{Q}^n$:

$$\begin{aligned} d(x,y) &= \text{number of positions in which vectors differ} \\ &= |\{i \in [n] : x_i \neq y_i\}| \end{aligned}$$

error-correcting code: $C \subseteq \mathcal{Q}^n$





d minimum distance

e error-correcting capacity

$$= \lfloor \frac{d-1}{2} \rfloor$$

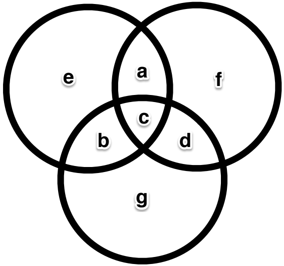
Linear code: $C \subseteq \mathbb{F}_q^n$ subspace of dimension k

Generator matrix: rows generate C

Encoding: $\mathbf{m}G = \mathbf{c}$

Parity check matrix: C is kernel of this matrix

$H\mathbf{c}^T = \mathbf{0}$



$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Typical problem:

Fix n and k (redundancy), make d as large as possible

Typical problem:

Fix n and k (redundancy), make d as large as possible

Singleton bound: $d \leq n - k + 1$

Equality: Maximum Distance Separable (MDS) code

Reed-Solomon code: pick $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$

$$C = \{(f(\alpha_1), \ldots, f(\alpha_n)) : f \in \mathbb{F}_q[x], \deg f < k\}$$

$$G = \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_{n-1} & \alpha_n \\ \vdots & \vdots & & \vdots & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_{n-1}^{k-1} & \alpha_n^{k-1} \end{pmatrix}$$

Reed-Solomon code is MDS

Several fast decoding algorithms known

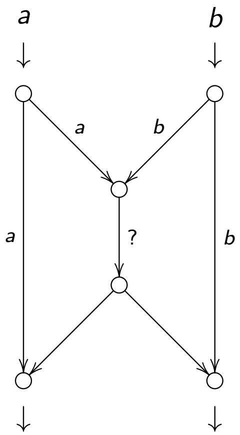
Needs large alphabet: $q > k$

Current day applications of error-correcting codes:

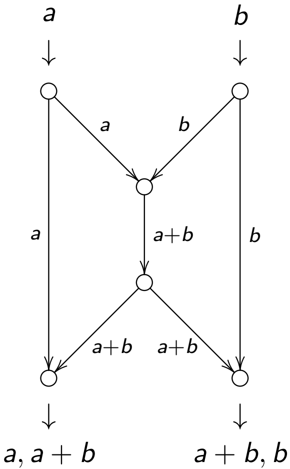
- Network coding

  Distributed storage

  Code-based crypto



a
b
a
b
a
?
b



Idea: send (rows of) matrices instead of vectors

Send: $X_1, \ldots, X_m \in \mathbb{F}_q^n$

Receive: $Y_1, \ldots, Y_m \in \mathbb{F}_q^n$

No errors: $Y = AX$

A full rank, known from the network structure

Send: $X_1, \ldots, X_m \in \mathbb{F}_q^n$

Receive: $Y_1, \ldots, Y_m \in \mathbb{F}_q^n$

No errors: $Y = AX$

A full rank, known from the network structure

In practice: $Y = A'X + Z$

A' rank erasures

Z errors

Send: $X_1, \ldots, X_m \in \mathbb{F}_q^n$

Receive: $Y_1, \ldots, Y_m \in \mathbb{F}_q^n$

No errors: $Y = AX$

A full rank, known from the network structure

In practice: $Y = A'X + Z$

A' rank erasures

Z errors

Decoding possible if $\mathrm{rk}(A')$ not too small and $\mathrm{rk}(Z)$ not too big.

Rank metric: $d(X, Y) = \mathrm{rk}(X - Y)$

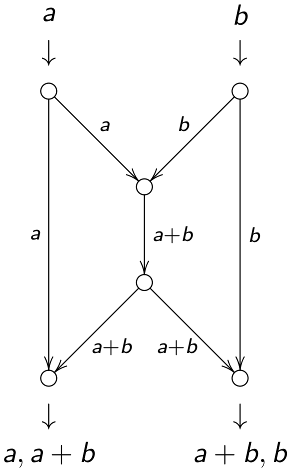
Depends on network structure

Well studied (Hui 1951, Delsarte 1978, Gabidulin 1995)

Good codes known

Ralf Kötter
(1963–2009)

Frank Kschischang
(*1962)



Better idea: send (bases of) subspaces instead of matrices

Random linear combinations

Send: basis of m-dim subspace $V \subseteq \mathbb{F}_q^n$

Receive: m vectors in $\mathbb{F}_q^n$

No errors: received vectors are basis of V

(with high probability)

Send: basis of m-dim subspace $V \subseteq \mathbb{F}_q^n$

Receive: m vectors in $\mathbb{F}_q^n$

No errors: received vectors are basis of V

(with high probability)

In practice: $U = \mathcal{H}_k(V) \oplus E$

$\mathcal{H}_k(V)$ random k-dim subspace of V

E error-subspace

Send: basis of m-dim subspace $V \subseteq \mathbb{F}_q^n$

Receive: m vectors in $\mathbb{F}_q^n$

No errors: received vectors are basis of V

(with high probability)

In practice: $U = \mathcal{H}_k(V) \oplus E$

$\mathcal{H}_k(V)$ random k-dim subspace of V

E error-subspace

Decoding possible if k not too small and $\dim(E)$ not too big.

Subspace distance: $d(U, V) = \dim(U) + \dim(V) - 2\dim(U \cap V)$

Independent of network structure

Faster transmission

Slower decoding

Few codes known

Current day applications of error-correcting codes:

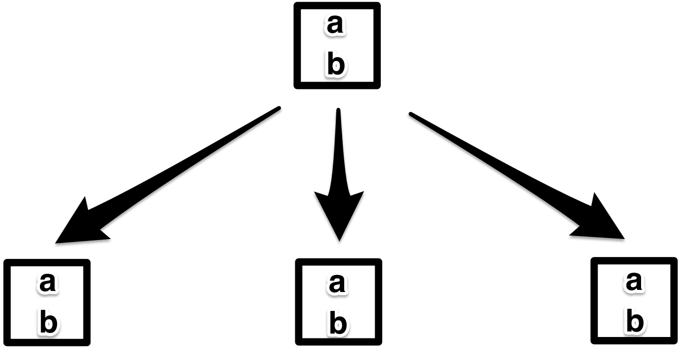
Network coding

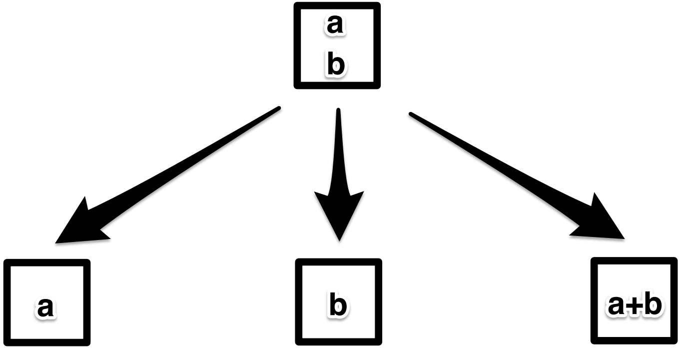
- Distributed storage

Code-based crypto

Google

facebook


a
b
a
b
a
b
a
b


a
b
a
b
a+b

Distributed storage demands different things from codes:

Erasures instead of errors

Small size: typically $n \leq 15$

Reed-Solomon codes do not preform well

Locality: minimize # nodes accessed during repair

Locality: minimize # nodes accessed during repair

Bandwidth: minimize total download bandwidth

Locality: minimize # nodes accessed during repair

Bandwidth: minimize total download bandwidth

Availability: optimize # repair possibilities

hot data vs. cold data

Current day applications of error-correcting codes:

Network coding

Distributed storage

- Code-based crypto

Public key cryptography

Everyone can encrypt with public function $\mathcal{E}$

Inverse of $\mathcal{E}$ (decryption) is hard to find

Only feasible with extra information about $\mathcal{E}$

Examples: factoring, DLP

Peter Shor
(*1959)

1994: algorithm for fast factoring using quantum computer

$\rightarrow$ post-quantum cryptography

Robert J. McEliece
(*1942)

Harald Niederreiter
(*1944)

McEliece crypto system (1978)

Private: Goppa code that can correct t errors

G generator matrix

S base change matrix

P permutation matrix

Public: scrambled generator matrix $G' = S \cdot G \cdot P$

McEliece crypto system (1978)

Private: Goppa code that can correct t errors

G generator matrix

S base change matrix

P permutation matrix

Public: scrambled generator matrix $G' = S \cdot G \cdot P$

Message $\mathbf{m}$, pick error vector $\mathbf{e}$ of weight at most t

Encryption: $\mathbf{m}G' + \mathbf{e}$

Decryption: decode received vector using S, P and G

Code-based crypto demands different things from codes:

Decoding random linear codes

Hidden structure

(Reed-Solomon codes are difficult to scramble)

Current day applications of error-correcting codes:

- Network coding
- Distributed storage
- Code-based crypto

Thank you for your attention.