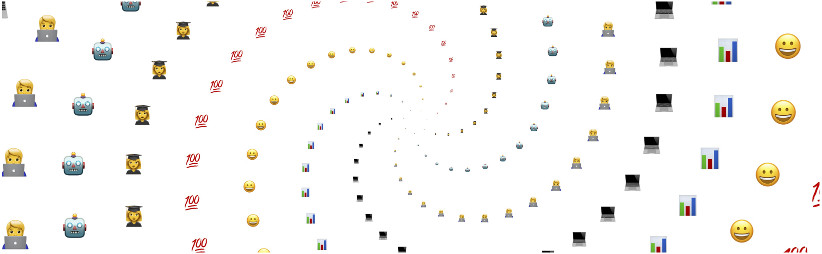
# Datenintegrität

## Kartentrick-Codierung

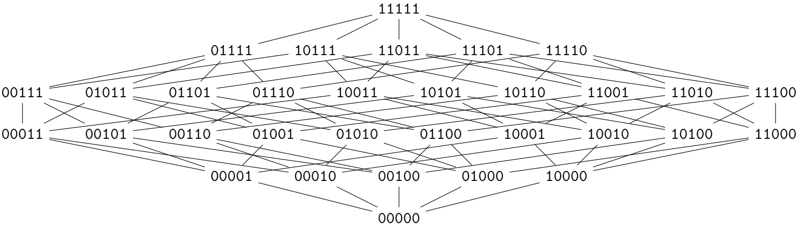
Cyril Wendl

Fachschaft Informatik
Kantonsschule im Lee



Kantonsschule Im Lee

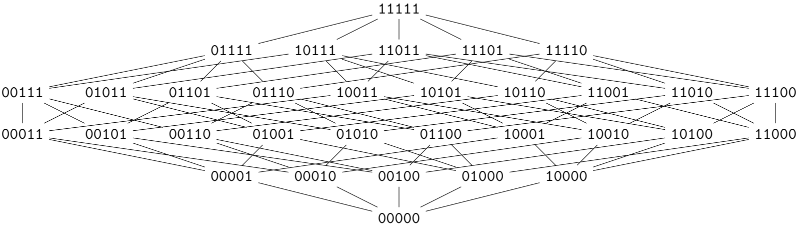
# Rückblick 1-fehlerkorrigierende Kodierung




Nachteil:

- Für Bitfolgen der Länge $n \to n$-dimensionaler Hyperwürfel

# Rückblick 1-fehlerkorrigierende Kodierung

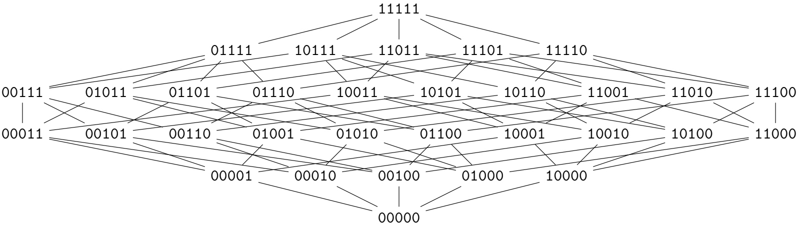



Nachteil:

- Für Bitfolgen der Länge $n \rightarrow n$-dimensionaler Hyperwürfel
- Anzahl Knoten? $\rightarrow 2^n$

Kantonsschule Im Lee

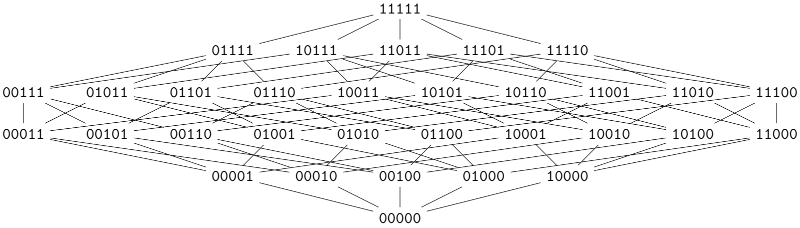
# Rückblick 1-fehlerkorrigierende Kodierung




Nachteil:

- Für Bitfolgen der Länge $n \to n$-dimensionaler Hyperwürfel
- Anzahl Knoten? $\to 2^n$
- Für längere Nachrichten wird der Hyperwürfel riesig

# Rückblick 1-fehlerkorrigierende Kodierung

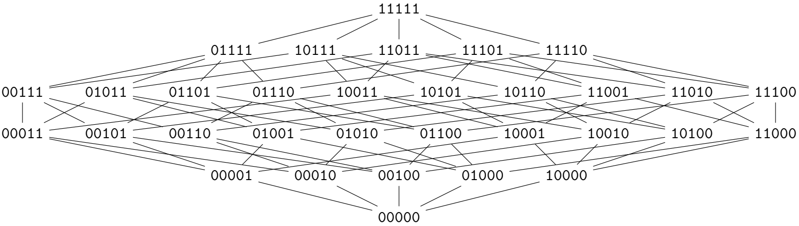



Nachteil:

- Für Bitfolgen der Länge $n \rightarrow n$-dimensionaler Hyperwürfel
- Anzahl Knoten? $\rightarrow 2^n$
- Für längere Nachrichten wird der Hyperwürfel riesig
- Aufwand wächst exponentiell!

Kantonsschule Im Lee

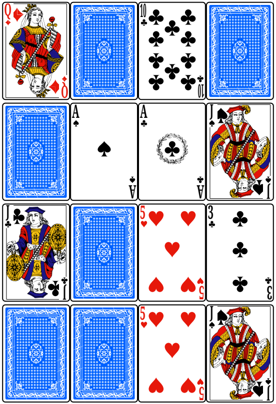
# Rückblick 1-fehlerkorrigierende Kodierung




Nachteil:

- Für Bitfolgen der Länge $n \to n$-dimensionaler Hyperwürfel
- Anzahl Knoten? $\to 2^n$
- Für längere Nachrichten wird der Hyperwürfel riesig
- Aufwand wächst exponentiell!
- Bessere Methode gesucht!

# „Zaubertrick“



**Erster Schritt**: Auslegen der ursprünglichen Karten ($n = m = 4$)

# „Zaubertrick“

**Zweiter Schritt**: Hinzufügen einer Reihe und einer Spalte (Hilfszauberer)

# „Zaubertrick“

**Dritter Schritt**: Karten, nachdem jemand eine Karte umgedreht hat

# „Zaubertrick“ (binär als Matrix)

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

**Erster Schritt**: Ursprüngliche Bitmatrix ($n = m = 4$)

# „Zaubertrick“ (binär als Matrix)

$$\begin{pmatrix} \underline{0} & \underline{1} & \underline{0} & \underline{1} & \underline{0} \\ 1 & 0 & 1 & 0 & \underline{0} \\ 0 & 1 & 1 & 1 & \underline{1} \\ 1 & 0 & 1 & 1 & \underline{1} \\ 0 & 0 & 1 & 1 & \underline{0} \end{pmatrix}$$

**Zweiter Schritt**: Hinzufügen einer Reihe und einer Spalte (<u>Kontrollbits</u>)

# „Zaubertrick“ (binär als Matrix)

$$\begin{pmatrix} \underline{0} & \underline{1} & \underline{0} & \underline{1} & \underline{0} \\ 1 & 0 & 1 & 0 & \underline{0} \\ 0 & 1 & 1 & 1 & \underline{1} \\ 1 & 0 & 0 & 1 & \underline{1} \\ 0 & 0 & 1 & 1 & \underline{0} \end{pmatrix}$$

**Dritter Schritt**: Eine Bitkarte wurde umgedreht (Fehlerfall)

# Die Kartentrickmethode

Für die folgende 6-stellige Bitfolge soll eine 1-fehlerkorrigierende Kodierung gefunden werden: `010101`

| | | |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 0 | 1 |

Ursprüngliche Nachricht

# Die Kartentrickmethode

Für die folgende 6-stellige Bitfolge soll eine 1-fehlerkorrigierende Kodierung gefunden werden: `010101`

$$\begin{array}{c|c|c} 0 & 1 & 0 \\ \hline 1 & 0 & 1 \end{array} \quad \longrightarrow \quad \begin{array}{c|c|c||c} \underline{1} & \underline{1} & \underline{1} & \underline{1} \\ \hline\hline 0 & 1 & 0 & \underline{1} \\ \hline 1 & 0 & 1 & \underline{0} \end{array}$$

Ursprüngliche Nachricht

Ursprüngliche Nachricht + <u>Kontrollbits</u>

# Optimalität: so wenig Kontrollbits wie möglich

Mehrere Möglichkeiten für Codewörter der Länge 16

- $16 = 4 \cdot 4 \rightarrow \underline{\text{Kontrollbits: } 4 + 4 + 1 = 9}$

# Optimalität: so wenig Kontrollbits wie möglich

Mehrere Möglichkeiten für Codewörter der Länge 16

- $16 = 4 \cdot 4 \rightarrow \underline{\text{Kontrollbits: } 4 + 4 + 1 = 9}$
- $16 = 2 \cdot 8 \rightarrow \underline{\text{Kontrollbits: } 2 + 8 + 1 = 11}$

# Optimalität: so wenig Kontrollbits wie möglich

Mehrere Möglichkeiten für Codewörter der Länge 16

- $16 = 4 \cdot 4 \rightarrow \underline{\text{Kontrollbits: } 4 + 4 + 1 = 9}$
- $16 = 2 \cdot 8 \rightarrow \underline{\text{Kontrollbits: } 2 + 8 + 1 = 11}$
- $16 = 1 \cdot 16 \rightarrow \underline{\text{Kontrollbits: } 1 + 16 + 1 = 18}$

# Optimalität: so wenig Kontrollbits wie möglich

Mehrere Möglichkeiten für Codewörter der Länge 16

- $16 = 4 \cdot 4 \to$ $\underline{\text{Kontrollbits: } 4 + 4 + 1 = 9}$
- $16 = 2 \cdot 8 \to$ $\underline{\text{Kontrollbits: } 2 + 8 + 1 = 11}$
- $16 = 1 \cdot 16 \to$ $\underline{\text{Kontrollbits: } 1 + 16 + 1 = 18}$
- Die beiden Seiten des Rechtecks sollten möglichst gleich gross sein

# Optimalität: so wenig Kontrollbits wie möglich

Mehrere Möglichkeiten für Codewörter der Länge 16

- $16 = 4 \cdot 4 \rightarrow$ $\underline{\text{Kontrollbits: } 4 + 4 + 1 = 9}$
- $16 = 2 \cdot 8 \rightarrow$ $\underline{\text{Kontrollbits: } 2 + 8 + 1 = 11}$
- $16 = 1 \cdot 16 \rightarrow$ $\underline{\text{Kontrollbits: } 1 + 16 + 1 = 18}$
- Die beiden Seiten des Rechtecks sollten möglichst gleich gross sein
- Bei Codewörtern der Länge 48 $\rightarrow 6 \cdot 8$ ist optimal!

# Auftrag (Skript)

- ✎ 1.20, 1.21

Falls bereits fertig:

- Kapitel 1.6 lesen und die Aufgaben 1.22 - 1.23 lösen