5.3 Übertragungssicherheit

🎯 Lernziele

Nach dieser Einheit sind Sie in der Lage dazu

- verschiedene Fehlertypen bei der Datenübertragung unterscheiden
- Maßnahmen zur Erkennung und Behebung von Übertragungsfehlern beschreiben
- die Hamming-Distanz verschiedener Codes ermitteln

🧠 Informationsgehalt von Telegrammen

Steuerfeld	Quelladresse	Zieladresse	Routing Z.	Länge	Nutzinformation	Sicherungs F.
1 Byte	2 Byte	2 Byte+1 Bit	3 Bit	4 Bit	1 Bit bis 14 Byte	1 Byte

- Unterscheidet sich je nach Bussystem, üblich sind unter anderem:
    - **Steuerfeld**: Priorität der Nachricht
    - **Quelladresse**: Absender (vgl. MAC-Adresse)
    - **Zieladresse**: Empfänger (vgl. MAC-Adresse)
    - **Routing Zähler**: Zählt wie oft über Koppler gesendet
    - **Nutzinformation**: Eigentlich Information (z.B. Messwerte eines Sensor)
    - **Sicherungs-Feld**: Wurden die Daten richtig übertragen (vgl. Hash)

Datensicherung

```
Gesendet:     0100001000000111000000
Empfangen 1:  0100001100000111000000
Empfangen 2:  010000100000011100000
```

- Bit können aus verschiedenen Gründen verloren gehen (z.B. Störung durch Elektromagnetische Felder, Probleme mit der Taktung, etc.)
- wie stellt man sicher, dass keine Daten **verloren** gehen oder **korrumpiert** werden?
  - OSI-Schicht 1: **technische Vorkehrungen** die Wahrscheinlichkeit von Störungen, z. B. durch geschirmte Kabel, Glasfaserkabel, potentialfreie Übertragung.
  - OSI-Schicht 2: **Überwachung** der Nachricht auf Fehler und Gegenmaßnahmen

🧠 Fehlerarten

- Wir betrachten im Folgenden stets transparente (**bitorientierte**) Codes. (d.h. jede Bitkombination ist erlaubt und sinnvoll)
- Bitfolge allein lässt nicht auf einen eventuellen Fehler schließen
- Es gibt drei Arten von Fehlern

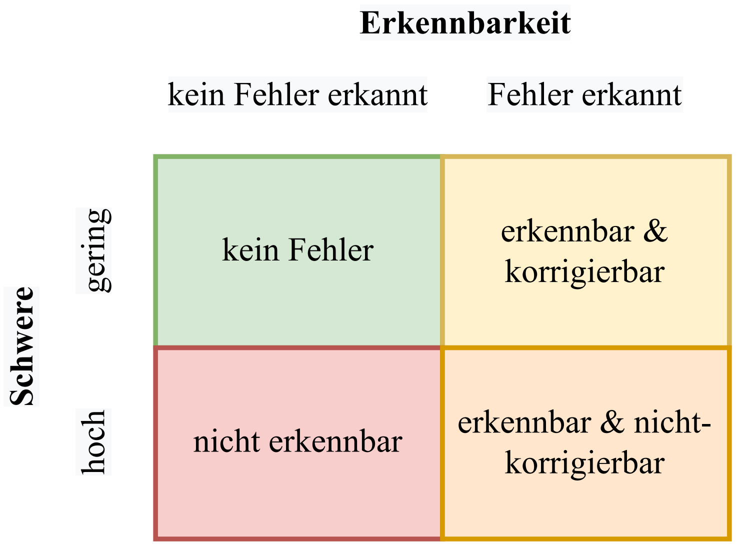
		Erkennbarkeit	
		kein Fehler erkannt	Fehler erkannt
Schwere	gering	kein Fehler	erkennbar & korrigierbar
	hoch	nicht erkennbar	erkennbar & nicht-korrigierbar

Fehlermaße

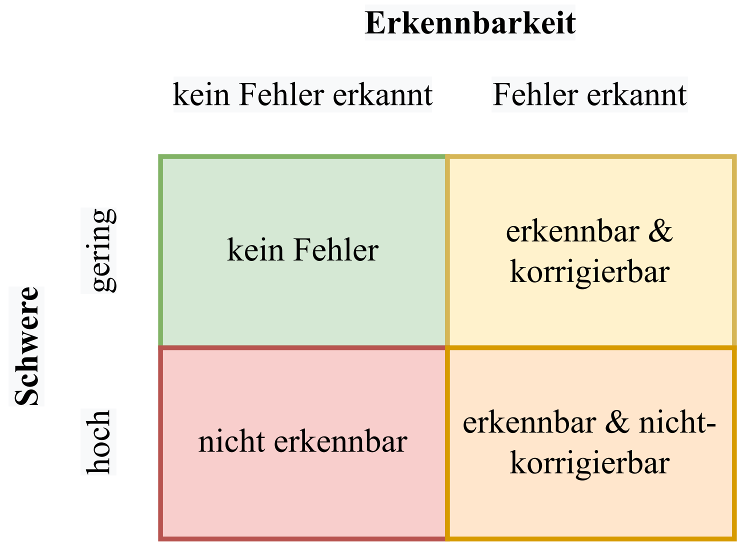
- **Bitfehlerrate** p
  $p = \frac{Anzahl\ der\ fehlerhaften\ Bits}{Gesamtzahl\ der\ gesendeten\ Bits}$
- Der ungünstigste Wert $p = 0.5$.
  Jedes zweite Bit ist dann im Mittel gestört, die Nachricht also wertlos
- wäre $p = 1$:
  - `001` : `110`
- realistischer Wert $p = 10^{-4}$

Erkennen von Übertragungsfehlern

- Ob Fehler erkennbar sind, hängt auch davon ab, wie die Information codiert wurde
- Code: z.B. Deutsche Sprache
  - Fehler ist offensichtlich: `Mein`, `Tein`
  - Fehler ist nicht erkennbar: `Mein`, `Dein`
  - Fehler ist erkennbar und korrigierbar: `Gxbäude`, `Gebäude`



- Codes können so definiert, werden, dass das Auftreten einzelner Übertragungsfehler offensichtlich wird.
    - `00`: Schalter **ein**
    - `01`: nicht definiert
    - `10`: nicht definiert
    - `11`: Schalter **aus**
    - Die Schalterstellung kann nicht verwechselt werden (bei einem Ein-Bit-Fehler)



Hamming-Abstand

- Unter dem Hamming-Abstand H eines Codes versteht man das **Minimum aller Abstände** zwischen verschiedenen Wörtern innerhalb des Codes
- **Abstand**: An wie vielen Stellen muss ein Wort verändert werden
- $H(\{00, 11\}) = 2$
- $H(\{00, 01, 10, 11\}) = 1$
- $H(\{00110, 00100\}) = 1$
- $H(\{'12345', '13349'\}) = 2$
- $H(\{'Haus', 'Baum', 'Tier'\}) = 2$

[Quelle](Beachte: bei den Strings zählt nicht, wie weit die Buchstaben auseinander liegen)

Anwendung des Hamming-Abstand zur Fehlererkennung

- Ein Code besteht aus folgenden drei Wörtern:
- `aus`, `ein`, `sie`
- Der kleinste der drei Abstände ist 2, also ist der Hamming-Abstand des Codes ebenfalls gleich $h = 2$ (zwischen `ein`, `sie`).
- Bei Codes mit Hamming-Abstand `h=2` **können alle `1`-Bit-Fehler erkannt werden.**
- D.h. der veränderte Code kann mit keinem anderen Wort verwechselt werden (`_ie`, `s_e`, `si_`)
- Ein `2`-Bit-Fehler kann nicht immer erkannt werden (`ein`, `_i_`, `sie`)

✍️ Aufgabe 5_3_1: Drehschalter

- Drehschalter **vier Einstellmöglichkeiten**
- werden als binäre Zahl (Codewort) an einen Empfänger übermittelt:
  - `00`, `01`, `10`, `11`
- Empfänger erhält das Codewort, hat sonst keine Möglichkeit, die Schalterstellung zu überprüfen

Quelle

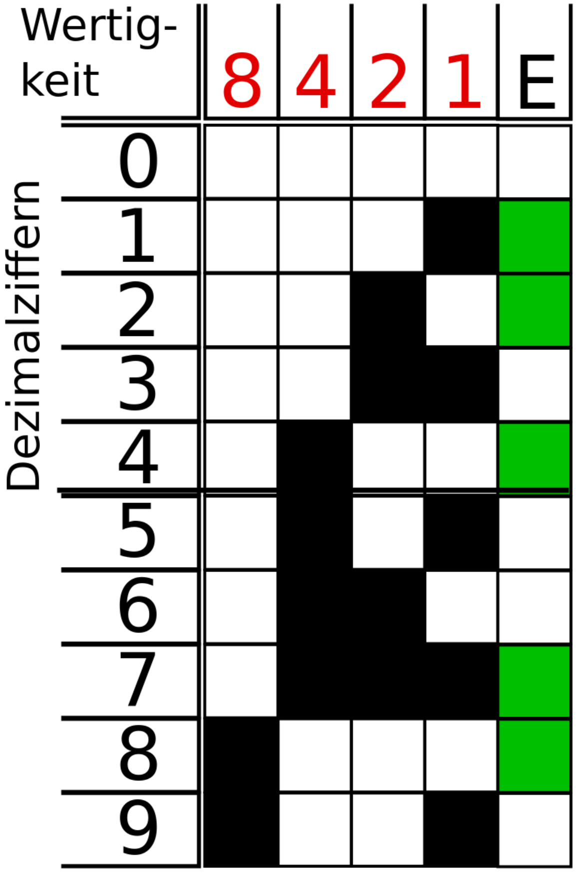
- `00`, `01`, `10`, `11`
- Hamming-Abstand zwischen den vier Worten ist jeweils 1,
- d. h. falls durch einen Fehler nur ein Bit umgekehrt wird, erhält der Empfänger zwar ein anderes, aber ebenso gültiges Codewort
    - Angenommen es treten nur Einfachfehler auf (es wird also maximal ein Bit geflippt)
    - Kann man einen binären Code entwickeln, der es nicht nur ermöglicht Fehler zu erkennen, sondern diese auch zu beheben?

✓ Lösung

- Um Einfachfehler zu korrigieren benötigt man einen Code, der einen Hamming-Abstand ≥ 3 hat:
  - z. B. `11000000` , `00110000` , `00001100` , `00000011` .
- Einfachfehler können nur erkannt und behoben werden:
  - `10000000` --> `11000000`
  - `11100000` --> `11000000`
  - `10110000` --> `00110000`

Paritätsbit zur Fehlererkennung

- Wir senden eine Zahl mit 4 Bit, z. B. `0010` (2_{10})
- Zahl der positiven Bits im Binärcode ist ungerade
- Paritätsbit `E=1` (even = True) wird hinzugefügt
  (Paritäts-/ Evenbit ist 1, wenn einegerade Zahl von Bit übertagen werden)
  und mit übertragen
- Alle ungeraden Anzahlen an Fehlern werden erkannt:
  - Original: `0010` - `E=1` - erwartet `E=1`
  - 1 Fehler: `0011` - `E=1` - erwartet `E=0`
  - 1 Fehler: `0010` - `E=0` - erwartet `E=1`
  - 2 Fehler: `1010` - `E=1` - erwartet `E=1`



Blocksicherung

- Anstelle nur nach allen X-Bits eine Paritätsbit einzufügen wird auch ein spaltenweises Paritätsbit

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

[Quelle](Gerhard Schnell & Bernhard Wiedemann)

1 Bit-Fehler in Daten

	1.	2.	3.	4.	5.	6.	7.	P
1.	0	0	0	1	0	1	1	1 0
2.	1	1	1	0	0	0	1	0
3.	1	0	0	1	0	0	1	1
4.	0	0	1	1	0	0	1	1
5.	1	1	0	0	1	1	1	1
6.	0	0	1	1	0	0	1	1
7.	1	1	0	0	0	1	1	0
P	0	10	1	0	1	1	1	0

Fehler

Kontrollrechnung

1 Bit-Fehler in Kontrollfeld

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

Fehler

Kontrollrechnung

2 Bit-Fehler in Kontrollfeld

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

Fehler

Kontrollrechnung

Unerkennbarer 4 Bit-Fehler in Daten und Kontrollfeld

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

Fehler

Weiterhin gerade Zahl an Paritätsbits

Kontrollrechnung

▶ 3Blue1Brown: A discovery-oriented introduction to error correction code