

Flagbyte mit Bytestopfen



Flagbyteverlust



Codierungsverletzung (Abstraktionsbruch)

Codierung auf Bitübertragungsebene verwendet nur ein Subset der Zeichen, z.B. für Taktrückgewinnung.

Beispiel: 4B/5B

Signalisierung über verbotene Zeichen; unproblematisch, da selten.

Zusammenfassung der abstrakten Grundlagen

Dienste

- Verbindungslos, unbestätigt
- Verbindungslos, bestätigt
- Verbindung, bestätigt, geordnet

Rahmen

- Längenfeld
- Flagbyte/-bit
- Codierungsverletzung

Bestätigung als Optimierung:
Früher korrigieren.

Arten von Fehlern

Welche Arten von Fehlern gibt es? Welche Struktur haben die Veränderungen?

- Verlorene Einzelbits
- Einzelne Bitflips
- Burst-Fehler (mehrere)

Hamming-Abstand

Codewörter

- 0000000000
- 0000011111
- 1111100000
- 1111111111

Korrigierbar:

- 0000000000
- 0000000011
- 0000011111

Erkennbar:

- 0000000000
- 0000001111
- 0000011111

Hamming-Abstand:

- 0000000000
- 0000011111
- 0000011111

p=95% korrekte Bits:

- HA 1: 40% beschädigt (1 / 3)
- HA 2: 8.6% beschädigt (1 / 12)
- HA 3: 1.2% beschädigt (1 / 87)
- Ha 5: 0.0096% beschädigt (1 / 10k)

Summe binom(10, p, 11-HA)

→ <https://de.wikipedia.org/wiki/Binomialverteilung>

Prüfbits Minimum, Einzelbitfehler

n gesamtbits, m Nachrichtenbits und r Prüfbits. $n = m + r$

m bits → 2^m mögliche Nachrichten → Pro Nachricht zusätzlich n verbotene Codewörter mit Abstand 1 nötig.

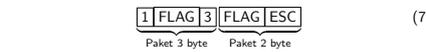
Verbotene Codewörter: Für jede erlaubte Nachricht jedes Bit flippen (⇒ n flips). ⇒ n + 1 Bitmuster für jede erlaubte Nachricht.

$$(n + 1) \cdot 2^m \leq 2^n = 2^{m+r} \quad (10)$$

$$(m + r + 1) \leq 2^r \quad (11)$$

$$m \leq 2^r - r - 1 \quad (12)$$

Flagbytes: Escaping nötig



Wie in Java-Strings:

String withQuote = "ab\"c";

Codierungsverletzung (Abstraktionsbruch)

Codierung auf Bitübertragungsebene verwendet nur ein Subset der Zeichen, z.B. für Taktrückgewinnung.

Beispiel: 4B/5B

Signalisierung über verbotene Zeichen; unproblematisch, da selten.

Aber Abstraktionsbruch:

Wechsel der Codierung erfordert Wechsel der Rahmenbildung.

Fehlerkorrektur oder -erkennung

Arten von Fehlern

- Erkennung vs. Korrektur
- Hamming-Abstand
- Fehlerkorrektur
- Fehlererkennung

Erkennung vs. Korrektur

Effizienzfrage:

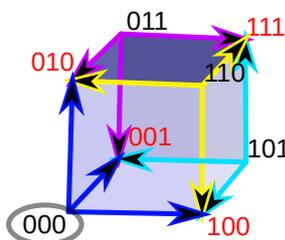
- Immer Zusätzliche Bandbreite
- Teure Neu-Übertragungen (Round-Trips) im Fehlerfall

Fehlerkorrektur, Methoden

Vollständige Suche

- Hamming-Code
- Binäre Faltungscodes
- Reed-Solomon-Codes
- LDPC-Codes (Low-Density Parity Check)

Verbotene Wörter, 1 bit erkennen



Flagbits mit Bitstopfen

FLAG 01111110
Escape x011111x -> x0111110x
Unescape x0111110x -> x011111x



Kombiniert

- Beispiel: Präambel + Längenfeld in WiFi (72 Bit Präambel!)

Frage: Wofür ist die Länge noch nützlich?

Arten von Fehlern

Welche Arten von Fehlern gibt es? Welche Struktur haben die Veränderungen?

Hamming-Abstand

Codewörter

- 0000000000
- 0000011111
- 1111100000
- 1111111111

p=95% korrekte Bits:

- HA 1: 40% beschädigt (1 / 3)
- HA 2: 8.6% beschädigt (1 / 12)
- HA 3: 1.2% beschädigt (1 / 87)
- Ha 5: 0.0096% beschädigt (1 / 10k)

Summe binom(10, p, 11-HA)

→ <https://de.wikipedia.org/wiki/Binomialverteilung>

Vollständige Suche

- Komplexität: O(N) (alle möglichen Codewörter vergleichen)

⇒ zu teuer.

⇒ Hinweise, wo der Fehler korrigiert werden kann.

