Universität
Paderborn
Fachgebiet
Rechnernetze

Wintersemester 2019/2029

Rechnernetze

Übung 5: Error detection, ARQ, Relationship FEC, CRC, ARQ

2019-11-21

---

1. **Fehlerdetektion**

   Benutzen Sie eine erweiterte Paritätstechnik, um einen fehlerdetektierenden Code zu entwerfen.

   - Eine Folge von $n = k \cdot l$ Bits wird hierbei zeilenweise in eine Matrix mit k Spalten und l Zeilen geschrieben.
   - In einer zusätzlichen Zeile wird für jede Spalte ein Paritätsbit berechnet. (z.B. even parity).
   - Die Daten werden zeilenweise übertragen.

   (a) Beschreiben Sie, wie der Empfänger eine solche Datenfolge auswertet.

   Lösung: Nach Empfang des gesamten Blockes werden die Paritätsbits geprüft. Ist mind. ein Paritätsbit falsch, so wird der gesamte Block für falsch erklärt.

   (b) Geben Sie ein Beispiel an, z.B. für $k = 3$ und $l = 4$.

   Lösung: Bitfolge, die übertragen werden soll 101110001011:

1	0	1
1	1	0
0	0	1
0	1	1
0	0	1

   Übertragen wird $101\ 110\ 001\ 011\ 001$ (Leerzeichen nur zur Orientierung)

   (c) Wie verhält sich das Verfahren bei einzelnen Bitfehlern oder bei “burst”-Bitfehlern? Wie lange kann eine durchgehende Folge von Bitfehlern sein, die erkannt wird?

   Lösung: Einzelne Bitfehler werden erkannt. Durchgehende Bitfehler werden bis zu einer Länge von k Bitfehlern erkannt, da pro Spalte nur ein Bit

verändert wird. Ein “Burst” der Länge $k+1$ wird genau dann nicht erkannt, wenn genau das erste und das letzte Bit im “Burst” falsch sind und alle anderen richtig, usw.

(d) Wie verändert sich das Verfahren, wenn Sie noch eine zusätzliche Spalte mit zeilenweise berechneten Paritätsbits hinzufügen? Überlegen Sie sich mögliche Vor- und Nachteile.

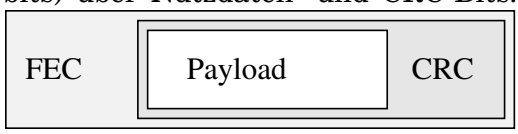
Lösung:

- Vorteile:
  - Es werden jetzt durchgehende Bitfehler der Länge k+1 gefunden.
  - Bei einzelnen Bitfehlern kann festgestellt werden, wo der Fehler ist, und kann dann korrigiert werden.
- Nachteile:
  - Es werden mehr Prüfbits benötigt, d.h. der Overhead steigt.
- $=>$ Abwägung zwischen Overhead im fehlerfreien Fall und kleineren Neuübertragungen im Einzelbitfehlerfall!

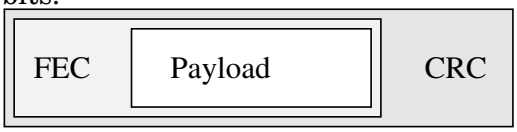
2. **Fehlererkennung und Fehlerkorrektur**

Das folgende Bild zeigt drei mögliche Szenarien, wie CRC und Vorwärtsfehlerkorrektur (FEC) kombiniert werden können:

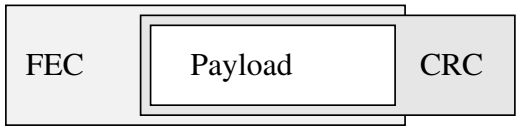
In *Szenario 1* berechnet der Sender zunächst die CRC-Prüfsumme zu den Nutzdaten (Payload). Anschließend berechnet er FEC-Bits (Fehlerkorrekturbits) über Nutzdaten- und CRC-Bits.



In *Szenario 2* berechnet der Sender zunächst FEC-Bits über die Nutzdaten. Anschließend berechnet er die CRC-Prüfsumme zu FEC- und Nutzdatenbits.



In *Szenario 3* werden FEC-Bits und CRC-Prüfbits ausschließlich über die Nutzdaten berechnet. Die Reihenfolge, in der die beiden Operationen am Sender ausgeführt werden, ist irrelevant.



Die Konkatenation aus FEC-Bits, Payload und CRC-Bits bildet die zu übertragende Nachricht.

(a) Welche Schritte muss der Empfänger in *Szenario 1* bzw. *Szenario 2* durchführen, nachdem er eine Nachricht empfangen hat?

Lösung: Alles genau umgekehrt wie beim Sender. Also: In Szenario 1 zuerst FEC auswerten, dann CRC. In Szenario 2 zuerst CRC auswerten, dann FEC. Für Szenario 2 ist es auch zulässig, wenn jemand angibt zuerst mit FEC Payload korrigieren, dann Konkatenation FEC, Payload wiederherstellen und erst dann darüber CRC auswertung machen.

(b) Sollte der Empfänger in *Szenario 3* zuerst die Fehlerkorrektur (FEC) durchführen oder die CRC-Prüfsumme überprüfen oder ist dies egal?

Lösung: Zuerst Fehlerkorrektur, weil dann ggf. Fehler in der Payload korrigiert werden können bevor die CRC-Überprüfung zu dem Schluss kommt dass ein Paketfehler vorliegt.

(c) Es trete ein einzelner Bitfehler **im CRC Teil** der Nachricht auf. In welchen Szenarien führt dies zu einer fehlerhaften CRC Überprüfung und somit zu einem Empfangsabbruch, auch wenn die Fehlerkorrektur (FEC) diesen Fehler korrigieren kann?

Lösung: Szenario 2 und Szenario 3, weil keine Fehlerkorrigierenden Redundanzbits für den CRC Teil vorhanden sind und somit FEC diesen Teil nicht korrigieren könnte.

(d) Es trete ein einzelner Bitfehler **im Payload Teil** der Nachricht auf. In welchen Szenarien führt dies zu einer fehlerhaften CRC Überprüfung und somit Verwerfen des Paketes, auch wenn die Fehlerkorrektur (FEC) diesen Fehler korrigieren kann?

Lösung: Szenario 2, weil zuerst CRC angewendet wird, dann theoretisch FEC. Da CRC Überprüfung fehlerhaft ist, wird abgebrochen und FEC kommt garnicht mehr zur Anwendung.

(e) Warum werden in *Szenario 2* die durch FEC und CRC eingefügten Redundanzbits deutlich schlechter ausgenutzt als in *Szenario 1*?

Lösung: In Szenario 2 im Falle eines Bitfehlers wird die Fehlerkorrektur nicht mehr durchgeführt, obwohl sie möglicherweise den Fehler beheben könnte. Dadurch sind die FEC Bits überflüssig. In Szenario 1 wird zuerst ggf. korrigiert und dann CRC ausgeführt. Dadurch werden alle Redundanzinformationen ausgewertet.

(f) Eine Übertragung war erfolgreich, wenn der Empfänger nach Auswertung der empfangenen Nachricht den selben Nutzdatenbitstring hat, den der Sender verschickt hat und wenn die CRC-Überprüfung erfolgreich war. Angenommen, es tritt ein einzelner Bitfehler an beliebiger Stelle auf. Ist dann in *Szenario 1* oder in *Szenario 3* die Wahrscheinlichkeit höher, dass die Übertragung erfolgreich war?

Lösung: In Szenario 1 kann ein einzelner Bitfehler potentiell immer behoben werden, egal wo er auftritt. In Szenario 3 kann er definitiv nicht behoben werden, falls er im CRC-Teil auftaucht. In diesem Fall führt der fehlschlagende CRC-Check zwangsläufig zu einer nicht-erfolgreichen Übertragung.

3. **Effizienz von ARQ** Wir betrachten einen Link Layer mit folgenden Annahmen:

   - Es werden Pakete der Länge l_P Bits und Acknowledgements der Länge l_A Bits verwendet.
   - Die Ausbreitungsverzögerung sei τ (in beide Richtungen gleich).
   - Die physikalische Datenrate in beiden Richtungen ist r bit/Sekunde.
   - Die Bitfehlerwahrscheinlichkeit ist in beiden Richtungen p, Bits werden von einander unabhängig beschädigt.
   - Es wird kein Vorwärtsfehlerschutz verwendet.
   - Der Übertragungskanal kann vollduplex.
   - Verarbeitungszeiten in Rechnern sind vernachlässigbar.

   (a) Bestimmen Sie die Effizienz (Anteil der Zeit, in der *neue* Daten gesendet werden) und die mittlere Datenrate bei Verwendung von Alternating Bit ARQ. Treffen Sie geeignete Annahmen für die Wahl des Timeouts.

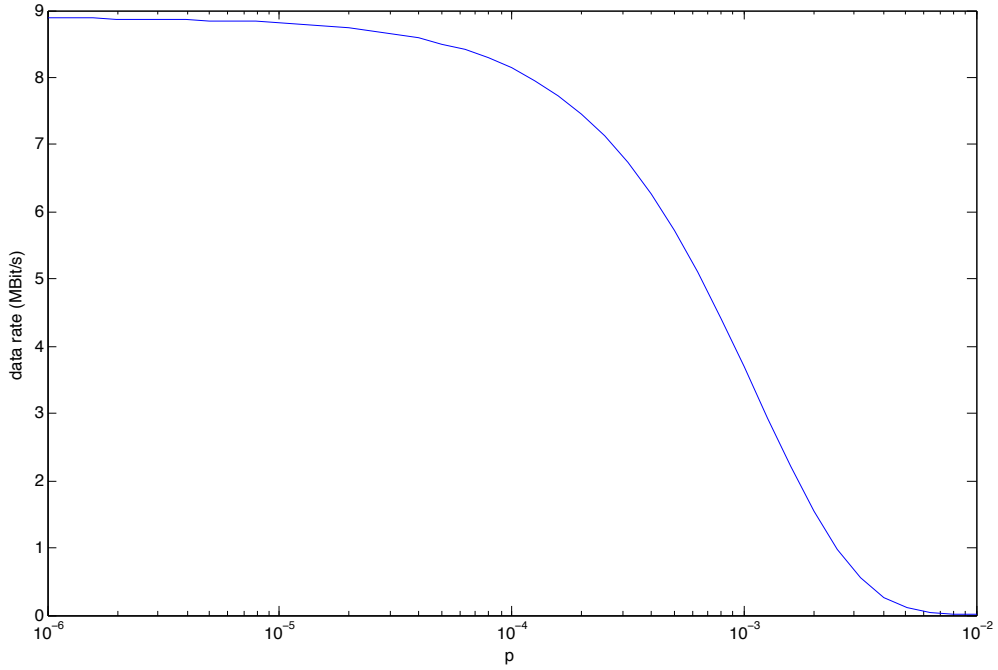
   Lösung:

   - Paketfehlerwahrscheinlichkeit: $P_P = 1 - (1-p)^{l_P}$ bzw. Fehlerwahrscheinlichkeit für Acknowledgements: $P_A = 1 - (1-p)^{l_A}$
   - Bestmöglicher Timeout: $T = (l_P/r + \tau) + (l_A/r + \tau) = \frac{l_P + l_A}{r} + 2\tau$
   - Timeout tritt ein, wenn entweder Paket verloren geht oder Paket erfolgreich ist und Acknowledgement verloren geht. $P_T = P_P + (1 - P_P)P_A$.
   - Damit ist mittlere Anzahl an Übertragungen für ein Paket $E[N] = \frac{1}{1-P_T}$ (N ist eine Zufallsvariable, die die Anzahl der Übertragungen beschreibt $\rightarrow$ Geometrische Verteilung, siehe Übung 1)
   - Damit ist Effizienz $E = \frac{1}{E[N]} \frac{l_P/r}{T}$, die mittlere effektive Datenrate ist rE.

     Erinnerung: Die Effizienz bezeichnet den Anteil der Zeit in der neue Nutzdaten gesendet werden.
     Anmerkung: Es gibt zwei Datenraten: die Physikalische (reine Übertragungen, mit/ohne Beachtung von Neuübertragungen) und die Effektive (nur neue Nutzdaten)

   (b) Plotten Sie die effektive Datenrate abhängig von der Bitfehlerwahrscheinlichkeit für folgende Werte:

$l_P = 100 \cdot 8$	100 Byte Paketlänge
$l_A = 10 \cdot 8$	10 Byte Ack-Länge
$r = 10\,000\,000$	10 MBit/s
$\tau = 0.1/1000$	0.1ms latenz



Lösung: