Einführung in CRC-Berechnungen

Alexander Köster

7. Oktober 2018

0.a) Wie funktioniert die CRC-Prüfung und welche Fehler können korrigiert werden?

Das CRC-Verfahren (*cyclic redundancy check*, „zyklische Redundanzprüfung“) ist eine Art „Prüfsumme“, ähnlich wie Hash-Funktionen, das einen Wert an alle übertragene Daten anhängt, damit der Empfänger prüfen kann, ob die Daten während der Übertragung beschädigt wurden.

Grundlage und Berechnung:
Die CRC betrachtet eine zu übertragende Bitfolge $b = (b_k b_{k-1} \dots b_2 b_1 b_0)_2$ der Länge k (d.h. ein Vektor im $\mathbb{F}_2$-Vektorraum $\mathbb{F}_2^k$) als Polynom über dem zweielementigen Körper $\mathbb{F}_2 := \{0, 1\}$, also als:

$$P_b(x) := \sum_{i=0}^{k} b_i x^i = b_k x^k + b_{k-1} x^{k-1} + \dots + b_2 x^2 + b_1 x + b_0 \in \mathbb{F}_2[x]$$

Für jede CRC-Anwendung ist außerdem ein festes Generatorpolynom $G(x) \in \mathbb{F}_2[x]$ vorgegeben.
Der Grad des Generatorpolynoms (höchste x-Potenz mit Koeffizient $\neq 0$) sei im Folgenden $n := \operatorname{Grad} G$.
Der CRC-Wert c als Polynom $P_c(x) \in \mathbb{F}_2[x]$ ist dann:

$$P_c(x) := P_b(x) \cdot x^n \bmod G(x)$$

Dieser Rest kann durch Polynomdivision berechnet werden; es handelt sich nach den Rechenregeln der Polynomdivision um ein Polynom des Grades $l < n$, also maximal $n - 1$, und die Koeffizienten können daher wieder als Bitfolge (Vektor in $\mathbb{F}_2^{n-1}$) aufgefasst werden.

(Das Auffassen von Vektoren als Polynome und umgekehrt ist begründet durch die Vektorraum-Isomorphie $K[x]_{\operatorname{Grad}<n} \cong K^n$ für jeden Körper K und jedes feste $n \in \mathbb{N}$.)

Die Polynomdivision in $\mathbb{F}_2[x]$ ist dabei besonders leicht. Man muss sich den Quotienten im Fall der CRC nicht merken, man muss für jede x-Potenz nur einen möglichen Koeffizienten testen (die 1) und es gilt $-1 = 1$, also ist jede Subtraktion eine einfache Addition.
Addition in $\mathbb{F}_2$ ist dabei für Computer ein `XOR` der Bits.

Aus dem entstandenen $P_c(x)$ kann man nun den zu senden Wert s berechnen, als Polynom $P_s(x) \in \mathbb{F}_2[x]$:

$$P_s(x) := P_b(x) \cdot x^n + P_c(x)$$

Fasst man dieses wieder als Vektor auf, so ist

$$s = (b_k b_{k-1} \dots b_1 b_0 c_{n-1} c_{n-2} \dots c_1 c_0) \in \mathbb{F}_2^{k+n-1}$$

die zu sendende Bitfolge mit CRC-Prüfwert.

Prüfen auf Empfängerseite:
Zum Prüfen muss nun der empfangene Wert $s' \in \mathbb{F}_2^{k+n-1}$ nur noch erneut mit Rest durch das gleiche Generatorpolynom $G(x)$ geteilt werden:

$$R(x) := P_{s'}(x) \bmod G(x)$$

Ist dann die Nachricht korrekt, so ist $R = 0$ (also das Nullpolynom).

Denn: Ist $s = s'$ (d.h. Nachricht korrekt) und damit $P_{s'} = P_s$ als Polynome in $\mathbb{F}_2[x]$, so gilt:

$$\begin{aligned}
R(x) &= P_s(x) \bmod G(x) \\
&= \left(P_b(x) \cdot x^n + P_c(x)\right) \bmod G(x) \\
&= \left(P_b(x) \cdot x^n \bmod G(x) + P_c(x) \bmod G(x)\right) \bmod G(x) \quad \text{(nach Modulo-Rechenregeln)} \\
&= (P_c(x) + P_c(x)) \bmod G(x), \text{ da } \operatorname{Grad} P_c(x) = l < \operatorname{Grad} G(x) = n \\
&= 0 \bmod G(x), \text{ da Elemente in } \mathbb{F}_2[x] \text{ (additiv) selbstinvers sind (Addition = Subtraktion)} \\
&= 0 \quad \text{(Nullpolynom)}
\end{aligned}$$

Umgekehrt stimmt dies aber nicht immer. Es kann Fälle geben, in denen die Nachricht falsch ist, aber das gleiche Restpolynom P_c rauskommt, und damit $R = 0$.
Bei gut gewähltem $G(x)$ kann dies aber sehr unwahrscheinlich gemacht werden.
Kommt nämlich nicht die korrekte Nachricht an, so gilt $s' = s + e$ mit $e \in \mathbb{F}_2^{k+n-1}$ als Fehler, und damit

$$R(x) = P_e(x) \bmod G(x) \quad \text{nach Modulo-Rechenregeln (analog oben).}$$

Folgende Fehler können dann bei jeweils bestimmten $G(x)$ erkannt werden:

- **Ein-Bit-Fehler**
  Ist nur ein einzelnes Bit (z.B. an der Stelle i) *flipped*, also e der i-te Einheitsvektor bzw. $P_e(x) = x^{i-1}$, so wird $R \neq 0$ genau dann, wenn $G(x)$ kein Monom ist (mehr als einen Term enthält).
- **Zwei Ein-Bit-Fehler**
  Hier ist $P_e(x) = x^i + x^j = x^i\left(x^{i-j} + 1\right)$ für $i < j$. Für x^i gilt dasselbe wie oben. Und bei einer guten Wahl von $G(x)$ teilt dieses nicht das Polynom $x^k + 1$ für alle k bis zu einer ausreichend großen Grenze (z.B. gesetzt als maximale Nachrichtenlänge), sodass $R \neq 0$ wird.
- **Ungerade Anzahl von Fehlern**
  Enthält $G(x)$ den Faktor $x + 1$, so wird bei einer ungeraden Parität (Anzahl von Termen) von $P_e(x)$ immer x oder 1 als Rest übrig bleiben.
- **Bündelfehler**
  Bei $m < n$ hintereinanderliegenden Fehlern (d.h. $P_e(x) = x^i\left(x^m + \cdots + x^2 + x + 1\right)$) ist immer $R \neq 0$, wenn $G(x)$ den Term 1 enthält, da G kein Polynom von kleinerem Grad als sich (Grad n) teilen kann.
- **Fehler im CRC-Wert** können natürlich auch auftreten. Diese sind aber unwahrscheinlicher als Fehler im Datenteil, da der CRC-Wert üblicherweise viel kleiner als der Datenteil ist.

Die CRC ist also nur ein fehlererkennender Code.
Es gibt auch einen zur CRC ähnlichen fehlerkorrigierenden Code (HEC/FEC), der 1-Bit-Fehler erkennen und korrigieren kann.

Quellen:

- *RFID-Handbuch: Grundlagen und praktische Anwendungen von Transpondern, kontaktlosen Chipkarten und NFC* von Klaus Finkenzeller, S.237ff.
- `http://www-users.math.umn.edu/~garrett/coding/Overheads/08_crcs.pdf`, S.8 und 10ff.
- `http://einstein.informatik.uni-oldenburg.de/rechnernetze/fehlerkorrektur1.htm`
- Vorlesungsreihe „(Lineare) Algebra“ von Prof. Mohamed Barakat

Beispiel:
Sei das Generatorpolynom $G(x) := x^5 + x^4 + x^2 + 1$ und die zu übertragenden Daten $b := 1101_2$.
Um Schreibarbeit zu sparen, kann man statt der ganzen Polynome auch nur ihre Koeffizientenfolgen in $\mathbb{F}_2$ schreiben. Dann kann man also $G(x)$ schreiben als 110101_2, sowie $P_b(x) \cdot x^n$ als 100100000_2.

Man kann nun die Polynomdivision durchführen, wobei wir uns wie oben erwähnt den Quotienten nicht merken und nur den subtraktiven (in $\mathbb{F}_2$ additiven) Teil durchführen:

```
 1101100000
+110101      <- bei der höchsten Potenz als Quotienten anfangen (= ganz links)
-------
 0000110000  <- XOR-Addition, Nullen runterholen
+    110101  <- bei der nächsten 1 weiterteilen (entspricht dem Wählen
     ------     der höchsten Potenz von x als Teil des Quotienten)
     000101  <- Rest
```

Der Rest ist hier also $c = 101_2$, bzw. Polynom $P_c(x) = x^2 + 1$, da der Grad nun kleiner ist als der von G.
Übertragen wird die Bitfolge:

```
 1101100000
+       101
-----------
 1101100101
```

und geprüft wird folgendermaßen:

```
 1101100101
+110101
 ------
     110101
+    110101
     ------
     000000
```

Durch den Rest 0 kann also mit hoher Wahrscheinlichkeit angenommen werden, dass die Nachricht korrekt ist (was sie in diesem Fall ist).

Hier kann man auch gut sehen, dass die Berechnung der CRC durch einen Computer oder durch Hardware sehr einfach ist, da es sich nur um XOR-Operationen auf Bitfolgen handelt.

Bei Bluetooth wird als Generatorpolynom das CRC-16 CRC-CCITT-Polynom $G(x) := x^{16} + x^{12} + x^5 + 1$ verwendet. Dieses erfüllt alle oben angegebenen Voraussetzungen für die verschiedenen Fehler.

Quellen:

- https://de.wikipedia.org/wiki/Zyklische_Redundanzpr%C3%BCfung
- https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx?doc_id=286439, S.455 (bzgl. PDF)