Die RWTH-Kundennummer

Guido Bunsen und Jürgen Müller

1 Warum eine RWTH-Kundennummer?

(1.1) Wie andere Organisationen, die viele Mitglieder, Angehörige oder Kunden zu verwalten haben, reicht es auch bei einer Hochschule von der Größe der Rheinisch-Westfälischen Technischen Hochschule (RWTH) Aachen nicht, den Namen einer Person zu wissen, um ein Zeugnis auszustellen, ein Buch auszuleihen, eine Einstellung oder eine Klausuranmeldung vorzunehmen. Bei mehreren zehntausend Studierenden, Alumni und Mitarbeitern ist die Gefahr zu groß, dem falschen 'Thomas Lehmann' oder der falschen 'Julia Schröder' eine Mahnung für ausgeliehene Bücher zu schicken, oder sie mit einem vorzeitigen Diplomzeugnis für das falsche Fach zu beglücken.

Eine Lösung wäre, den Namen nicht alleine zu verwenden, sondern immer nur in Kombination etwa mit der Anschrift, dem Geburtsort und dem Geburtsdatum. Allerdings soll nicht bei jedem Verwaltungsvorgang all diese Information offengelegt werden müssen. Daher verwenden Organisationen in der Regel Kundennummern oder Mitgliedsnummern, um eine Person identifizieren zu können. Leider nutzen praktisch alle Organisationen verschiedene Methoden, um die Nummern zuzuweisen und deren Eindeutigkeit sicherzustellen. Das gilt nicht nur für die Vielzahl von Verwaltungen, wie etwa den Rentenversicherungen, Meldebehörden, Krankenversicherungen, Finanzämtern oder Automobilclubs, sondern bereits innerhalb einzelner Organisationen, wie etwa der RWTH:

Das Studierendensekretariat vergibt Matrikelnummern; die Personalstelle vergibt Personalnummern; die RWTH-Bibliothek vergibt Nummern auf Leseausweisen; und das Rechen- und Kommunikationszentrum vergibt Benutzerkennungen. Wie zu erwarten sind diese Nummern unabhängig voneinander, und häufig hat eine Person jede dieser vier Nummern: Viele Studierende stehen in einem Angestelltenverhältnis mit der RWTH, und nutzen sowohl die Bibliothek als auch die Dienste des Rechen- und Kommunikationszentrums.

(1.2) Daher erscheinen die nachfolgenden Anforderungen an eine einheitliche RWTH-Kundennummer, eine 'RWTH-ID', wie sie im folgenden genannt werden soll, sinnvoll:

i) Für die nächsten Jahrzehnte sollen ausreichend viele Kundennummern zur Verfügung stehen, ohne alte Nummern erneut verwenden zu müssen.

ii) In die Bildung der Kundennummern sollen keine personenbezogenen Informationen einfließen, oder daraus ablesbar sein.

iii) Die Kundennummern sollen so dargestellt sein, daß sie sich leicht merken lassen und leicht als RWTH-ID erkannt werden.

iv) Einfache Übermittlungs- oder Tippfehler sollen erkannt werden.

Die erste Forderung läßt sich recht leicht konkretisieren: Bei durchschnittlichen Studierendenzahlen von 30 000 und einer Verweildauer von etwa 5 Jahren sind jedes Jahr etwa 6 000 neue Kundennummern für die Studierenden zu vergeben. Die Zahl der Mitarbeiter, Gastwissenschaftler und Projektpartner wird diese Zahl wohl nicht erreichen. Geht man in der Summe von 10 000 neuen pro Jahr zu vergebenen Kundennummern aus, so läßt sich direkt erkennen, daß eine Zahl von 1 000 000 verschiedenen Kundennummern für eine längere Zukunft völlig ausreichend sein wird.

Bislang ist es häufig üblich, Kundennummern fortlaufend zu vergeben. Etwa bei Matrikelnummern wird so verfahren, aber in diesem Fall läßt sich das Eintrittsdatum erkennen, was man vielleicht Dritten gegenüber nicht offenlegen möchte. Im Falle der Verwendung von Informationen aus dem Namen tritt bei Namensänderung regelmäßig der Wunsch auf, auch die Kundennummer möge sich ändern. Um der zweiten Forderung also weitestgehend nachzukommen, ist es daher naheliegend, die Kundennummer zufällig zu bilden.

Bei der Darstellung der Kundennummern gibt es natürlich eine Vielzahl von Möglichkeiten. Zunächst ist zu entscheiden, aus welchem Symbolen die RWTH-ID gebildet werden soll: Will man sich lediglich auf die zehn Dezimalziffern beschränken, so benötigt man sechs Ziffern um 1 000 000 verschiedene Kundennummern zur Verfügung zu haben. Um der vierten Forderung Genüge zu tun, sollten zu diesen Ziffern dann noch eine oder zwei Prüfziffern kommen. Damit erhält man eine Kundennummer aus sieben oder acht Ziffern, die man sich wohl nur schwer merken kann.

Bildet man die Kundennummer jedoch aus den 26 Buchstaben des Alphabets unter Einbeziehung von Groß- und Kleinschreibung und der zehn Ziffern, so hat man 62 Symbole zur Verfügung. Damit kann man schon mit vier Symbolen $26^4 = 14\,776\,336$, also fast 15 Millionen, Kundennummern bilden. Allerdings hat auch dieser Zeichenvorrat Nachteile: Beim Buchstabieren muß man immer hinzufügen, ob es sich um einen kleinen oder großen Buchstaben handelt, und beim Lesen ist oft nur schwer zu unterscheiden, ob man den Buchstaben O oder die Ziffer 0 vor sich hat.

Es gibt also gute Gründe, sich auf einen kleineren Symbolvorrat zu beschränken. Für die RWTH-ID wurde entschieden, unter Einbeziehung von Ziffern und Großbuchstaben 32 verschiedene alphanumerische Symbole zu verwenden; Kleinbuchstaben und Symbole, die leicht verwechselt werden, finden keine Verwendung. In (1.3) wird deutlich werden, warum gerade 32 verschiedene Symbole günstig sind. Damit kann man mit vier Symbolen $32^4 = 1\,048\,576$ Kundennummern bilden, mit fünf Symbolen bereits $32^5 = 33\,554\,432$, eine komfortable Anzahl. Kommt noch ein Prüfsymbol dazu, so erhält man eine RWTH-ID, die aus sechs dieser Symbole besteht.

Eine weitere Frage betrifft die Benutzerfreundlichkeit: Zwar ist die Verwendung von Kürzeln und Nummern an sich keine benutzerfreundliche Maßnahme, aber dennoch kann man die Verwendung erleichtern, indem man für einen guten

Wiedererkennungswert sorgt. Dabei helfen eine einheitliche Schreibweise, die immer verwendet wird, wenn eine RWTH-ID auf Bescheinigungen, Anträgen, Ausweiskarten oder Internetseiten ausgegeben wird, und eine konstante Länge und die Gruppierung der einzelnen Symbole. Man kennt solche Verfahren etwa von Kreditkarten, auf denen die Kontonummer in Vierergruppen dargestellt wird. Im Fall der RWTH-ID werden immer genau sechs Symbole verwendet, wobei zwei Gruppen von je drei Zeichen durch einen Bindestrich voneinander getrennt werden: etwa SL8-BRX.

(1.3) Ein weiterer Aspekt beim Design der RWTH-ID ist die Fehlererkennung: Die bei weitem häufigsten Tippfehler bei Tastatureingaben sind einzelne falsche Symbole oder das Vertauschen zweier benachbarter verschiedener Symbole; eine Statistik dazu findet man etwa in [3, Ch.1.2]. Die Prüfsymbole sollen daher so gewählt werden, daß diese Fehler erkannt werden. Der Kunde kann in solch einem Fall dann aufgefordert werden, seine RWTH-ID neu einzugeben.

Zur Fehlererkennung wird bei der RWTH-ID folgender Weg beschritten: Zunächst werden die verwendeten Symbole in Folgen von fünf Binärziffern, also 0 oder 1, übersetzt. Da es gerade $2^5 = 32$ solcher Binärfolgen gibt, erklärt dies auch die Anzahl der ausgewählten Symbole. Da sich die Symbole I und J nur wenig von 1, das Symbol O nur wenig von 0, und das Symbol V nur wenig von U unterscheiden, sollen I und J sowie O und V wegen besserer Lesbarkeit nicht verwendet werden. Die zugelassen Symbole und ihre Übersetzungen in Binärfolgen der Länge 5 sind in Abbildung 1 aufgelistet.

Mit dieser Übersetzung besteht eine RWTH-ID intern also aus einer Binärfolge der Länge 30, die alphanumerischen Symbole werden nur zur Eingabe und zur Ausgabe benutzt. Für SL8-BRX erhält man etwa

$$[11001 \mid 10011 \mid 01000 \mid 01011 \mid 11000 \mid 11101],$$

wobei wir hier zur besseren Lesbarkeit Trennstriche eingefügt haben. Auf diese Binärfolgen kann man nun Methoden der mathematischen Codierungstheorie anwenden, die den Weg weisen werden, wie man Prüfsymbole geschickt wählt, um die oben beschriebenen Tippfehler erkennen zu können. Außerdem werden wir Wert darauf legen, daß die Ergebnisse theoretischen Aanalyse sich in effiziente und sehr kurze Programme umsetzen lassen, die nur maschinennahe Bitoperationen verwenden.

2 Polynome

Im folgenden beschreiben wir kurz den abstrakten mathematischen Hintergrund, den wir brauchen, um mit Binärfolgen rechnen zu können. Wir beschränken uns auf die Begriffe und Aussagen, die in Abschnitt 3 benutzt werden, um konkret zu beschreiben, wie die RWTH-ID gebildet wird. Mehr über die zugrundeliegende Algebra und Computeralgebra findet man etwa in [4, Ch.4] und [2, Sect.I.2, I.3].

Abbildung 1: Übersetzung alphanumerischer Symbole in Binärfolgen.

0	00000
1	00001
2	00010
3	00011
4	00100
5	00101
6	00110
7	00111

8	01000
9	01001
A	01010
B	01011
C	01100
D	01101
E	01110
F	01111

G	10000
H	10001
K	10010
L	10011
M	10100
N	10101
P	10110
Q	10111

R	11000
S	11001
T	11010
U	11011
W	11100
X	11101
Y	11110
Z	11111

(2.1) Binärzahlen. Zunächst kann man so mit den Zahlen 0 und 1 rechnen, sie also addieren oder multiplizieren, daß jeweils wieder eine der Zahlen 0 oder 1 herauskommt: Man rechnet herkömmlich wie mit ganzen Zahlen, dividiert das Ergebnis dann aber durch 2, und nimmt den Rest dieser Division als Ergebnis. Am einfachsten stellt man dies durch die zugehörigen Additions- und Multiplikationstabellen dar:

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Dann gelten die üblichen Rechenregeln, wie etwa Kommutativität $a+b=b+a$ und $a\cdot b=b\cdot a$, oder das Distributivgesetz $(a+b)\cdot c=(a\cdot c)+(b\cdot c)$, weiter. Faßt man diese Verknüpfungen übrigens als Bitoperationen auf, so sind die Addition gerade eine exor-Operation und die Multiplikation eine and-Operation.

(2.2) Polynome. Um nun mit auch Binärfolgen $[a_0, a_1, a_2, \ldots, a_d]$ der Länge $d+1$ zu rechnen, interpretiert man sie als Summen der Form

$$f := a_0 \cdot X^0 + a_1 \cdot X^1 + a_2 \cdot X^2 + \cdots + a_d \cdot X^d,$$

die wir auch $f = \sum_{i=0}^{d} a_i X^i$ schreiben. Ausdrücke dieser Form werden als **Polynome** in der **Unbestimmten** X bezeichnet. Dies ist eine rein formale Übersetzung, die uns aber in (2.3) nahelegen wird, wie man mit diesen Objekten sinnvoll rechnen, sie also addieren oder multiplizieren kann. Da es bei allen Rechnungen dabei nur auf die Einträge der zugrundeliegenden Binärfolgen ankommen wird, können diese wieder als Bitoperationen interpretiert und implementiert werden.

Zur Vereinfachung schreiben wir X^i für $1 \cdot X^i$, und X für X^1. Wie sich in (2.3) zeigen wird, können X^0 mit 1, und $0 \cdot X^i$ mit 0 identifiziert werden. Daher können Summanden der Form $0 \cdot X^i$ beliebig weggelassen oder hinzugefügt werden. Wählt man in der Darstellung $f = \sum_{i=0}^{d} a_i X^i$ eines Polynoms $f \neq 0$ die obere Summationsgrenze d so klein wie möglich, was also $a_d = 1$ impliziert, so heißt die nicht-negative ganze Zahl d der **Grad** von f; wir schreiben $\mathrm{Grad}(f) = d$. Dem Nullpolynom ordnen wir $\mathrm{Grad}(0) := -\infty$ zu.

(2.3) Addition und Multiplikation. Es seien $f = \sum_{i=0}^{d} a_i X^i$ und $g = \sum_{i=0}^{d} b_i X^i$ Polynome, wobei man nötigenfalls durch Hinzufügen von Summanden der Form $0 \cdot X^i$ annehmen kann, daß f und g die gleiche Anzahl von Summanden haben. Dann definiert man die Addition durch

$$f + g = \left(\sum_{i=0}^{d} a_i X^i\right) + \left(\sum_{i=0}^{d} b_i X^i\right) := \sum_{i=0}^{d} (a_i + b_i) X^i,$$

wobei die Additionen $a_i + b_i$ gemäß (2.1) erfolgen.

Aus dieser Formel folgt insbesondere, daß $f + g = g + f$ gilt. Für die Grade der beteiligten Polynome hat man $\mathrm{Grad}(f + g) \leq \max\{\mathrm{Grad}(f), \mathrm{Grad}(g)\}$, wobei im Falle $\mathrm{Grad}(f) \neq \mathrm{Grad}(g)$ sogar $\mathrm{Grad}(f + g) = \max\{\mathrm{Grad}(f), \mathrm{Grad}(g)\}$ gilt.

Die Summenschreibweise von Polynomen deutet schon an, daß für die Multiplikation die Rechenregel $X^i \cdot X^j = X^{i+j}$ sowie das Distributivgesetz gelten sollen. Daher definiert man die Multiplikation durch

$$f \cdot g := \sum_{i=0}^{d} \sum_{j=0}^{d} a_i b_j X^{i+j} = \sum_{i=0}^{2d} \left(\sum_{j=\max\{0,i-d\}}^{\min\{i,d\}} a_j b_{i-j} \right) X^i.$$

wobei die Additionen und Multiplikationen der a_i und b_j wieder gemäß (2.1) erfolgen. Ausgeschrieben ist also

$$f \cdot g = (a_0 b_0) \cdot X^0 + (a_0 b_1 + a_1 b_0) \cdot X^1 + (a_0 b_2 + a_1 b_1 + a_2 b_0) \cdot X^2 + \cdots + (a_d b_d) \cdot X^{2d}.$$

Aus dieser Formel folgt insbesondere, daß $1 \cdot f = f$ und $f \cdot g = g \cdot f$ gelten, und daß für ein Produkt genau dann $f \cdot g \neq 0$ ist, wenn für beide Faktoren $f, g \neq 0$ gilt. Für die Grade der beteiligten Polynome hat man $\mathrm{Grad}(f \cdot g) = \mathrm{Grad}(f) + \mathrm{Grad}(g)$.

(2.4) Division. Die für uns wichtigste Eigenschaft von Polynomen ist, daß sie analog zu den ganzen Zahlen eine **Division mit Rest** erlauben: Sind f und $g \neq 0$ Polynome, so gibt es Polynome q, r mit $\mathrm{Grad}(r) < \mathrm{Grad}(g)$ und $f = q \cdot g + r$. Dabei heißen q der **Quotient** und r der **Rest** der Division.

Quotient und Rest sind eindeutig in folgendem Sinne: Sind q', r' auch Polynome mit $\mathrm{Grad}(r') < \mathrm{Grad}(g)$ und $f = q' \cdot g + r'$, so gilt $q = q'$ und $r = r'$: Angenommen, es ist $r' \neq r$. Dann ist $(q - q') \cdot g = r' - r \neq 0$, was wegen $\mathrm{Grad}((q - q') \cdot g) \geq \mathrm{Grad}(g)$ und $\mathrm{Grad}(r' - r) < \mathrm{Grad}(g)$ ein Widerspruch ist. Also ist $r = r'$, und wegen $(q - q') \cdot g = r' - r = 0$ und $g \neq 0$ folgt auch $q = q'$. ♯

Zur von Division mit Rest gibt es den in Abbildung 2 angegebenen einfachen Algorithmus. Es ist zu zeigen, daß dieser Algorithmus terminiert und den Quotienten und den Rest der Division zurückgibt: Bei einem Aufruf von (3) seien dazu $r = \sum_{i=0}^{e} a_i X^i$ und $g = \sum_{i=0}^{d} b_i X^i$, wobei $e = \mathrm{Grad}(r)$ und $d = \mathrm{Grad}(g)$. Dann hat

$$\begin{aligned} r + X^{e-d} \cdot g &= \textstyle\sum_{i=0}^{e} a_i X^i + \sum_{i=0}^{d} b_i X^{e-d+i} \\ &= \textstyle\sum_{i=0}^{e-d-1} a_i X^i + \sum_{i=e-d}^{e} (a_i + b_{d-e+i}) X^i, \end{aligned}$$

Abbildung 2: Polynomdivision in der Theorie.

(0) Man setzt $q := 0$ und $r := f$.
(1) Man unterscheidet zwei Fälle:
(2) Ist $\mathsf{Grad}(r) < \mathsf{Grad}(g)$, so terminiert man und gibt q und r zurück.
(3) Ist $\mathsf{Grad}(r) \geq \mathsf{Grad}(g)$, so ersetzt man

$$\begin{array}{rcl} q & \leftarrow & q + X^{\mathsf{Grad}(r)-\mathsf{Grad}(g)} \\ r & \leftarrow & r + X^{\mathsf{Grad}(r)-\mathsf{Grad}(g)} \cdot g \end{array}$$

und geht zu (1).

---

den e-ten Koeffizienten $a_e + b_d = 1 + 1 = 0$. Also wird bei jedem Aufruf von (3) der Grad von r verkleinert, daher tritt schließlich der Fall (2) ein, und der Algorithmus terminiert mit $\mathrm{Grad}(r) < \mathrm{Grad}(g)$. Außerdem gilt die Gleichung $f = q \cdot g + r$ in (0), und daher auch nach jeder Ausführung von (3). ♯

(2.5) Beispiel. Es seien $f = 1+X^2+X^4+X^6+X^8$ und $g = 1+X^2+X^5$, also ist $\mathrm{Grad}(g) = 5$. Dann erhält man bei vier Aufrufen von (3) der Reihe nach:

i	q_i	r_i	e_i
1	0	$1 + X^2 + X^4 + X^6 + X^8$	8
2	X^3	$1 + X^2 + X^3 + X^4 + X^5 + X^6$	6
3	$X + X^3$	$1 + X + X^2 + X^4 + X^5$	5
4	$1 + X + X^3$	$X + X^4$	4

Übersetzt man dies zurück in Binärfolgen, der Länge 10 etwa, so erhält man

$$f \longleftrightarrow [1010101010] \quad \text{und} \quad g \longleftrightarrow [1010010000].$$

Die Binärfolge zu einem Produkt $X^k \cdot g$ entsteht aus derjenigen zu g durch eine Verschiebung um k Stellen nach rechts. Die Addition zweier Polynome wird durch komponentenweise Addition der zugehörigen Binärfolgen ausgeführt. Daher erhalten wir die folgenden Binärfolgen für $r_1, \ldots, r_4$:

$$\begin{array}{rcl} r_1 & \longleftrightarrow & [1010101010] \\ +X^3 \cdot g & \longleftrightarrow & [000\underline{1010010}] \\ \hline r_2 & \longleftrightarrow & [1011111000] \\ +X \cdot g & \longleftrightarrow & [0\underline{101001}000] \\ \hline r_3 & \longleftrightarrow & [1110110000] \\ +g & \longleftrightarrow & [\underline{1010010}000] \\ \hline r_4 & \longleftrightarrow & [0100100000] \end{array}$$

(2.6) Teilbarkeit. Division mit Rest motiviert unmittelbar die folgenden Definitionen: Es seien $f, g \neq 0$ Polynome. Hat f bei Division durch g den Rest

0, so heißt g ein **Teiler** von f; wir schreiben $g \mid f$. Es gilt also genau dann $g \mid f$, wenn es ein Polynom q mit $f = q \cdot g$ gibt. Ist das Polynom 1 der einzige gemeinsame Teiler von f und g ist, so heißen f und g **teilerfremd**.

In Abschnitt 3 werden wir benutzen, daß $f = \sum_{i=0}^{d} a_i X^i$ genau dann teilerfremd zu X^j ist, wobei $j > 0$, wenn $a_0 \neq 0$ gilt. Daraus folgt sofort, daß f genau dann teilerfremd zu X^j ist, wenn f teilerfremd zu X ist. Die erste Aussage kann wie folgt eingesehen werden:

Ist $a_0 \neq 0$, so gilt $X \mid f$ und $X \mid X^j$. Weiter zeigt die Multiplikationsformel in (2.3), daß X^j genau von den Polynomen X^i, wobei $0 \leq i \leq j$ ist, geteilt wird. Unter diesen ist im Falle $a_0 = 0$ nur $X^0 = 1$ ein Teiler von f. ♯

(2.7) Teilerfremdheit. Außerdem werden wir In Abschnitt 3 folgende allgemeine Charakterisierung von Teilerfremdheit benötigen: Polynome $f, g \neq 0$ sind genau dann teilerfremd, wenn es Polynome s, t gibt mit $s \cdot f + t \cdot g = 1$. Zum Beweis argumentieren wir wie folgt:

Es seien zunächst s, t Polynome mit $s \cdot f + t \cdot g = 1$. Angenommen, es gibt ein Polynom h mit $\mathrm{Grad}(h) > 0$ und Polynome f', g' mit $f = h \cdot f'$ und $g = h \cdot g'$. Dann gilt $h \cdot (s \cdot f' + t \cdot g') = 1$, was wegen $\mathrm{Grad}(h \cdot (s \cdot f' + t \cdot g')) \geq \mathrm{Grad}(h) > 0$ ein Widerspruch ist. Also sind f, g teilerfremd.

Nun seien umgekehrt f, g teilerfremd. Dann setzt man $r_0 := f$ und $r_1 := g$, sowie für $i \geq 2$ mittels Division mit Rest der Reihe nach $r_{i-2} = q_{i-1} \cdot r_{i-1} + r_i$, wobei $\mathrm{Grad}(r_i) < \mathrm{Grad}(r_{i-1})$. Dies kann man so lange wiederholen, bis wegen der Gradbedingung schließlich $r_{l+1} = 0$ wird, für ein geeignetes $l \geq 1$. Dann ist r_l ein Teiler von r_{l-1}, und der Reihe nach folgert man, daß r_l auch ein Teiler von $r_{l-2}, r_{l-3}, \ldots, r_1, r_0$ ist, also ist $r_l = 1$. Außerdem gilt $r_0 = 1 \cdot f + 0 \cdot g$ und $r_1 = 0 \cdot f + 1 \cdot g$, und der Reihe nach folgert man, daß für $i \geq 2$ auch $r_i = s_i \cdot f + t_i \cdot g$ ist, für geeignete Polynome s_i, t_i. ♯

Damit hat man übrigens auch einen Algorithmus, auch **Euklidischer Algorithmus** genannt, um festzustellen, ob Polynome f, g teilerfremd sind oder nicht: Man iteriert Division mit Rest wie oben beschrieben, und überprüft, ob $r_l = 1$ oder $r_l \neq 1$ ist.

3 Codierung

Nun können wir genau beschreiben, wie die RWTH-ID gebildet wird. Mit der Übersetzung in (1.3), bei der jedem alphanumerischen Symbol eine Binärfolge der Länge $m = 5$ zugeordnet wird, betrachten wir also Binärfolgen der Länge $n = 30$, die wiederum nach (2.2) als Polynome f mit $\mathrm{Grad}(f) < n$ aufgefaßt werden können. Da wir Prüfinformation einfließen lassen wollen, gibt es **zulässige** und **unzulässige** Polynome mit $\mathrm{Grad}(f) < n$, also solche, die zu einer gültigen RWTH-ID gehören, und solche, die dies nicht tun. Unsere Aufgabe ist es somit, anhand der Diskussion in Abschnitt 1 geeignet festzulegen, welche Polynome zulässig sein sollen. In (3.5) erläutern wir die nachfolgenden Betrachtungen anhand des konkreten Beispiels SL8-BRX aus (1.3).

(3.1) Zulässige Polynome. Man wählt ein geeignetes **Erzeugerpolynom** g mit $\mathrm{Grad}(g) = k$, wobei $0 < k < n$ ist, und läßt genau diejenigen Polynome f mit $\mathrm{Grad}(f) < n$ zu, die bei Division durch g den Rest 0 haben. Weiter unter werden wir $k = 5$ wählen und noch weitere Bedingungen an das Polynom g stellen. Doch zunächst gehen wir den Fragen nach, wieviele zulässige Polynome es gibt, und wie man sie bestimmen kann:

Es seien zunächst $f = \sum_{i=0}^{n-1} a_i X^i$ ein beliebiges Polynom mit $\mathrm{Grad}(f) < n$, und nach Division mit Rest sei $f = q \cdot g + r$ mit $\mathrm{Grad}(r) < k$; also ist genau dann f zulässig, wenn $r = 0$ ist. Nun seien $f' := \sum_{i=0}^{k-1} a_i X^i$ und $f'' := \sum_{i=0}^{n-k-1} a_{k+i} X^i$, also ist $f = f' + X^k \cdot f''$. Damit ist $X^k \cdot f'' = q \cdot g + (r + f')$ mit $\mathrm{Grad}(r + f') < k$, also hat $X^k \cdot f''$ bei Division durch g den Rest $r + f'$. Also ist f genau dann zulässig, wenn $X^k \cdot f''$ bei Division durch g den Rest f' hat.

Somit erhält man jedes zulässige Polynom f mit $\mathrm{Grad}(f) < n$ genau einmal, wenn man ein beliebiges Polynom $f'' = \sum_{i=0}^{n-k-1} a_{k+i} X^i$ mit $\mathrm{Grad}(f'') < n - k$ wählt, den Rest $f' = \sum_{i=0}^{k-1} a_i X^i$ der Division von $X^k \cdot f''$ durch $g = \sum_{i=0}^{k} b_i X^i$ berechnet, und $f := f' + X^k \cdot f'' = \sum_{i=0}^{n-1} a_i X^i$ setzt. Daher gibt es genau so viele zulässige Polynome f mit $\mathrm{Grad}(f) < n$, wie es Polynome f'' mit $\mathrm{Grad}(f'') < n - k$ gibt. Von denen gibt es wiederum genau so viele, wie es Binärfolgen der Länge $n - k$ gibt, und deren Anzahl schließlich ist 2^{n-k}.

In der Praxis führen wir bei der Erzeugung von f aus f'' die Multiplikation $X^k \cdot f''$, die ja lediglich eine Verschiebung der Binärfolge zu f'' um k Stellen nach rechts bedeutet, und die Verschiebungen der Binärfolge zu g, wie in (2.5) beschrieben, nicht explizit durch, sondern verwenden eine effiziente Implementation, die wie in Abbildung 3 angedeutet mit möglichst wenigen Bitoperationen auskommt: Wir erhalten die Binärfolge zu f' als Ergebnis von $\mathsf{poldiv}([a_k, \ldots, a_{n-1}], n - k, [b_0, \ldots, b_{k-1}], k)$.

(3.2) Verifikation. Um nun umgekehrt von einem Polynom f mit $\mathrm{Grad}(f) < n$ festzustellen, ob es zulässig ist oder nicht, muß man nach (3.1) den Rest r von f bei Division durch g berechnen, und prüfen ob $r = 0$ oder $r \neq 0$ ist. Um dazu wiederum das Programm in Abbildung 3 zu benutzen, das ja bei Eingabe von f den Rest der Division von $X^k \cdot f$ durch g berechnet, geht man wie folgt vor:

Nach Division mit Rest sei $f = q \cdot g + r$ mit $\mathrm{Grad}(r) < k$, dann ist $X^k \cdot f = (X^k \cdot q) \cdot g + (X^k \cdot r)$. Weiter sei nach Division mit Rest $X^k \cdot r = q' \cdot g + r'$ mit $\mathrm{Grad}(r') < k$, also ist insbesondere $\mathrm{Grad}(q') = \mathrm{Grad}(X^k \cdot r) - \mathrm{Grad}(g) < k$. Außerdem ist $X^k \cdot f = (X^k \cdot q + q') \cdot g + r'$, somit hat $X^k \cdot f$ bei Division durch g den Rest r'. Daher kann r' wie oben beschrieben berechnet werden.

Unter der zusätzlichen **Voraussetzung**, daß g teilerfremd zu X sei, zeigen wir nun, daß genau dann $r = 0$ gilt, wenn $r' = 0$ ist: Ist $r = 0$, so ist auch $X^k \cdot r = 0$ und somit $r' = 0$. Nun sei umgekehrt $r' = 0$ und angenommen, es ist $r \neq 0$. Es seien s, t Polynome mit $s \cdot g + t \cdot X^k = 1$. Dann ist $t \cdot (q' \cdot g) = t \cdot (X^k \cdot r) = (1 + s \cdot g) \cdot r$, also $(t \cdot q' + s \cdot r) \cdot g = r \neq 0$, was wegen $\mathrm{Grad}((t \cdot q' + s \cdot r) \cdot g) \geq \mathrm{Grad}(g) = k$ ein Widerspruch ist.

Abbildung 3: Polynomdivision in der Praxis.

```
/* binseq: Binärfolge der Länge n */
/* genseq: Binärfolge der Länge k */

poldiv(binseq,d,genseq,k) {
    local rem = (genseq and 0);      /* Binärfolge der Länge k */
    for (i = 0; i < n; i++) {
        if (rem[k-1] exor binseq[n-1]) {
            rem >>= 1;
            rem = (rem exor genseq);
        } else {
            rem >>= 1;
        };
        binseq >>= 1;
    }
    return rem;
}
```

(3.3) Fehlererkennung. Wie bereits in (1.3) bemerkt, sind die bei weitem häufigsten Tippfehler bei Tastatureingaben **i)** einzelne falsche Symbole oder **ii)** das Vertauschen zweier benachbarter verschiedener Symbole. Wir zeigen nun, daß bei geeigneter Wahl von g beide Fehlertypen erkannt werden, also zu unzulässigen Polynomen führen. Wir nehmen an, daß die Symbole in Binärfolgen der Länge $m \leq k$ übersetzt werden, also ist insbesondere m ein Teiler von n; gemäß der Übersetzung in (1.3) werden wir unten $m = 5$ wählen. Wir nehmen gemäß (3.2) weiter an, daß g teilerfremd zu X ist.

Bei einem Tippfehler liegt also statt eines zulässigen Polynoms f ein Polynom $\widetilde{f} \neq f$ vor. Der Tippfehler wird genau dann erkannt, wenn $\widetilde{f}$ bei Division durch g einen Rest $r \neq 0$ hat. Nun hat aber das zulässige Polynom f bei Division durch g den Rest 0, also hat das **Fehlerpolynom** $\widehat{f} := \widetilde{f} + f \neq 0$ bei Division durch g ebenfalls den Rest r. Wir bestimmen die Fehlerpolynome obiger Tippfehler:

i) Ist das j-te Symbol falsch, wobei $1 \leq j \leq \frac{n}{m}$, so ist die dem richtigen Symbol zugeordnete Binärfolge durch eine falsche ersetzt. Also gibt es ein Polynom $h = \sum_{i=0}^{m-1} a_i X^i \neq 0$, so daß

$$\widehat{f} = \sum_{i=0}^{m-1} a_i X^{m(j-1)+i} = X^{m(j-1)} \cdot h.$$

Angenommen, es ist $r = 0$, dann ist $\widehat{f} = q \cdot g$ für ein q geeignet. Da g teilerfremd zu X ist, gibt es Polynome s, t mit $s \cdot g + t \cdot X^{m(j-1)} = 1$. Damit ist $t \cdot (q \cdot g) = t \cdot (X^{m(j-1)} \cdot h) = (1 + s \cdot g) \cdot h$, also $(t \cdot q + s \cdot h) \cdot g = h \neq 0$, was wegen $\mathrm{Grad}((t \cdot q + s \cdot h) \cdot g) \geq \mathrm{Grad}(g) = k \geq m > \mathrm{Grad}(h)$ ein Widerspruch ist. ♯

ii) Sind das j-te und das $(j+1)$-te Symbol verschieden und werden sie vertauscht,

wobei $1 \leq j < \frac{n}{m}$, so gibt es ein Polynom $h = \sum_{i=0}^{m-1} a_i X^i \neq 0$, so daß

$$\widehat{f} = \left(\sum_{i=0}^{m-1} a_i X^{m(j-1)+i} \right) + \left(\sum_{i=0}^{m-1} a_i X^{mj+i} \right) = X^{m(j-1)} \cdot (1 + X^m) \cdot h.$$

Angenommen, es ist $r = 0$, dann ist $\widehat{f} = q \cdot g$ für ein q geeignet. Da g teilerfremd zu X ist, gibt es Polynome s, t mit $s \cdot g + t \cdot X^{m(j-1)} = 1$. Setzt man $q' := t \cdot q + s \cdot h$, so erhält man $q' \cdot g = (1 + X^m) \cdot h$ analog zu (i).

Unter der zusätzlichen **Voraussetzung**, daß g teilerfremd zu $1 + X^m$ sei, gibt es Polynome s', t' mit $s' \cdot g + t' \cdot (1 + X^m) = 1$, und man erhält $t' \cdot (q' \cdot g) = t' \cdot ((1 + X^m) \cdot h) = (1 + s' \cdot g) \cdot h$, also $(t' \cdot q' + s' \cdot h) \cdot g = h \neq 0$, was wegen $\mathrm{Grad}((t' \cdot q' + s' \cdot h) \cdot g) \geq \mathrm{Grad}(g) = k \geq m > \mathrm{Grad}(h)$ ein Widerspruch ist. ♯

Wir schließen also: Wenn man für g neben der Teilerfremdheit zu X noch Teilerfremdheit zu $1 + X^m$ voraussetzt, so werden die obigen Tippfehler erkannt.

(3.4) Wahl des Erzeugerpolynoms. Für die RWTH-ID haben wir gemäß (1.3) $n = 30$ und $m = 5$. Da wir $32^5 = 2^{25}$ Kundennummern zur Verfügung stellen wollen, wählen wir gemäß (3.1) $k = 5$. Dann bestimmt man alle Erzeugerpolynome $g = \sum_{i=0}^{5} b_i X^i$, die die in (3.3) gemachten Voraussetzungen erfüllen, wie folgt:

Wegen $\mathrm{Grad}(g) = 5$ ist $b_5 = 1$, und da g teilerfremd zu X ist, ist $b_0 = 1$. Also hat man insgesamt $2^4 = 16$ Möglichkeiten, $b_1, \ldots, b_4$ zu wählen. Mit dem in (2.7) beschriebenen Euklidischen Algorithmus kann man dann aus diesen Kandidaten diejenigen herausfinden, die tatsächlich teilerfremd zu $1 + X^5$ sind. Diese Rechnungen kann man leicht mit dem Computeralgebra-System GAP [1] durchführen, und man erhält die in Abbildung 4 genannten Möglichkeiten für g. Die zweite Spalte gibt übrigens eine Zerlegung des jeweiligen Polynoms als Produkt von Polynomen vom Grad > 0 an, sofern es eine gibt; auch dies kann man leicht mit GAP berechnen. Für die RWTH-ID wählen wir aus diesen, im obigen Sinne gleich guten Polynomen das folgende Erzeugerpolynom:

$$\boxed{g := 1 + X^2 + X^5}$$

(3.5) Beispiel. Für das Beispiel SL8-BRX aus Abschnitt 1 geht man also wie folgt vor: Man wählt zunächst zufällig eine Folge von fünf Symbolen, also etwa L8BRX. Mit der Übersetzung in (1.3) erhalten wir

$$[10011 \mid 01000 \mid 01011 \mid 11000 \mid 11101],$$

und mit (2.2) daraus das Polynom

$$f'' := 1 + X^3 + X^4 + X^6 + X^{11} + X^{13} + X^{14} + X^{15} + X^{16} + X^{20} + X^{21} + X^{22} + X^{24}.$$

Division durch $g = 1 + X^2 + X^5$ ergibt $X^5 \cdot f'' = q \cdot g + (1 + X + X^4)$, wobei wir den Quotienten q hier nicht explizit benötigen. Nun setzt man $f' := 1 + X + X^4$ und

Abbildung 4: Erzeugerpolynome vom Grad 5.

g	Faktorisierung
$1+X+X^5$	$(1+X+X^2)\cdot(1+X^2+X^3)$
$1+X^4+X^5$	$(1+X+X^2)\cdot(1+X+X^3)$
$1+X^2+X^5$	
$1+X^3+X^5$	
$1+X+X^2+X^3+X^5$	
$1+X+X^2+X^4+X^5$	
$1+X+X^3+X^4+X^5$	
$1+X^2+X^3+X^4+X^5$	

$f := f' + X^5 \cdot f''$. Zum Polynom f' gehört die Binärfolge [11001], zurückübersetzt erhält man in der Tat das Symbol S. Zum Polynom f gehört die Verkettung der Binärfolgen zu f' und zu f'', also genau die Binärfolge aus (1.3).

Zur Verifikation von SL8-BRX nach (3.2) betrachtet man formal die Verkettung von [00000] mit der Binärfolge zu f, also

$$X^5 \cdot f \longleftrightarrow [00000 \mid 11001 \mid 10011 \mid 01000 \mid 01011 \mid 11000 \mid 11101];$$

das zugehörige Polynom $X^5 \cdot f$ hat in der Tat bei Division durch g den Rest 0. Wir betrachten nun Auswirkungen der in (3.3) betrachteten Tippfehler:

i) Wird in SL8-BRX statt des vierten Symbols B ein N getippt, so erhält man SL8-NRX mit zugehöriger Binärfolge

$$\widetilde{f} \longleftrightarrow [11001 \mid 10011 \mid 01000 \mid \underline{10101} \mid 11000 \mid 11101].$$

Das Fehlerpolynom $\widehat{f} = \widetilde{f} + f$ gehört damit zu folgender Binärfolge:

$$\begin{array}{rcl} \widetilde{f} & \longleftrightarrow & [11001 \mid 10011 \mid 01000 \mid \underline{10101} \mid 11000 \mid 11101] \\ +f & \longleftrightarrow & [11001 \mid 10011 \mid 01000 \mid \underline{01011} \mid 11000 \mid 11101] \\ \hline \widehat{f} & \longleftrightarrow & [00000 \mid 00000 \mid 00000 \mid \underline{11110} \mid 00000 \mid 00000] \end{array}$$

Also ist $\widehat{f} = X^{15} \cdot (1 + X + X^2 + X^3)$ und in der Tat

$$\widehat{f} = (X^2 + X^7 + X^9 + X^{11} + X^{12} + X^{13} +) \cdot g + (X^2 + X^4).$$

ii) Werden in SL8-BRX das vierte Symbol B und das fünfte Symbol R vertauscht, so erhält man SL8-RBX mit zugehöriger Binärfolge

$$\widetilde{f} \longleftrightarrow [11001 \mid 10011 \mid 01000 \mid \underline{11000} \mid \underline{01011} \mid 11101].$$

Das Fehlerpolynom $\widehat{f} = \widetilde{f} + f$ gehört damit zu folgender Binärfolge:

$$\begin{array}{rcl} \widetilde{f} & \longleftrightarrow & [11001 \mid 10011 \mid 01000 \mid \underline{11000} \mid \underline{01011} \mid 11101] \\ +f & \longleftrightarrow & [11001 \mid 10011 \mid 01000 \mid \underline{01011} \mid \underline{11000} \mid 11101] \\ \hline \widehat{f} & \longleftrightarrow & [00000 \mid 00000 \mid 00000 \mid \underline{10011} \mid \underline{10011} \mid 00000] \end{array}$$

Also ist $\widehat{f} = X^{15} \cdot (1 + X^5) \cdot (1 + X^3 + X^4)$ und in der Tat

$$\widehat{f} = (1 + X + X^3 + X^6 + X^{11} + X^{13} + X^{16} + X^{18} + X^{19}) \cdot g + (1 + X + X^2).$$

(3.6) Mathematischer Kontext. Abschließend kommentieren wir noch kurz den Kontext aus der mathematischen Codierungstheorie, auf dem die Realisierung der RWTH-ID beruht:

Die in (3.1) und (3.2) beschriebene Erzeugung und Verifikation von Binärfolgen wird auch als **CRC-Codierung** bezeichnet. Sie zeichnet sich durch einfache Codier- und Decodieralgorithmen aus, die sich sehr effizient implementieren lassen. Mehr dazu findet man etwa in [3, Ch.2] und [5, Ch.7].

Die Erkennung von **Fehlerbündeln** der Länge 5 und gewisser Fehlerbündel der Länge 10 in (3.3) beruht auf der Tatsache, daß CRC-Codes nahe verwandt mit **zyklischen Codes** sind. Da das hier gewählte Erzeugerpolynom $g = 1+X^2+X^5$ irreduzibel, also ein Teiler von $X^{31} + 1$ ist, bilden die zulässigen Binärfolgen einen nicht-zyklischen binären $[30, 25]$-Code, der durch **einfaches Verkürzen** aus dem von g erzeugten zyklischen binären $[31, 26]$-**Reed-Solomon-Code** entsteht. Mehr dazu findet man etwa in [3, Ch.3.4] und [5, Ch.10.2].

Literatur

[1] The GAP Group: GAP-4.4 — Groups, Algorithms, and Programming, 2006, http://www.gap-system.org.

[2] J. von zur Gathen, J. Gerhard: Modern computer algebra, second edition, Cambridge University Press, 2003.

[3] D. Jungnickel: Codierungstheorie, Spektrum Verlag, 1995.

[4] S. Lang: Algebra, revised third edition, Graduate Texts in Mathematics 211, Springer, 2002.

[5] F. MacWilliams, N. Sloane: The theory of error-correcting codes, North-Holland Mathematical Library 16, North-Holland, 1986.

G.B.:
Rechen- und Kommunikationszentrum, RWTH Aachen
Seffenter Weg 23, D-52074 Aachen
bunsen@rz.rwth-aachen.de

J.M.:
Lehrstuhl D für Mathematik, RWTH Aachen
Templergraben 64, D-52062 Aachen
Juergen.Mueller@math.rwth-aachen.de