1. Syndrom-Decodierung

Codierungstheorie
Anton Malevich
12.09.2017

Aufgabe 1

- Zeigen Sie, dass der binäre Code, der aus den folgenden Codewörtern besteht, 1-fehlerkorrigierend ist:

```
0000000
1111111
0001110
1110001
0010111
1101000
0011001
1100110
0100101
1011010
0101011
1010100
0110010
1001101
0111100
1000011
```

Aufgabe 2

Wir wollen einen binären Code konstruieren, der es uns erlaubt, die 26 Buchstaben des lateinischen Alphabets zu übertragen, und der zwei Fehler korrigieren kann. Um die 26 Buchstaben codieren zu können, brauchen wir ein Code der Dimension mindestens 5. Um zwei Fehler korrigieren zu können, muss die Minimaldistanz des Codes auch mindestens 5 sein. Die kleinste Länge, die das erlaubt, ist 13.

Sei darum C der folgende lineare Code mit Erzeugermatrix

In[3]:=
$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix};$$

Die Kontrollmatrix finden wir als

In[5]:=
```
H = NullSpace[G, Modulus → 2];
MatrixForm[H]
```

Out[6]//MatrixForm=

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- Warum?
- Zeigen Sie, dass dim $C = 5$ und $d(C) = 5$ ist.

  Wir schreiben alle Codewörter des Codes C in die Liste L

In[44]:=
```
Prepend[Rest@Subsets[G], {Array[0 &, 13]}];
L = Mod[Plus @@@ %, 2];
```

Wir benutzen die ersten 26 Codewörter, um die 26 Buchstaben zu codieren. Das erste Codewort entspricht dem Buchtaben A, das zweite dem B, usw., das 26-te dem Z.

In[151]:=
```
MapThread[#1 ↔ #2 &, {Alphabet[], L[[;; 26]]}];
TableForm@%
```

Out[152]//TableForm=
```
a ↔ {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
b ↔ {1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0}
c ↔ {0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1}
d ↔ {0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0}
e ↔ {0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1}
f ↔ {0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1}
g ↔ {1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1}
h ↔ {1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0}
i ↔ {1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1}
j ↔ {1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1}
k ↔ {0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1}
l ↔ {0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0}
m ↔ {0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0}
n ↔ {0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1}
o ↔ {0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1}
p ↔ {0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0}
q ↔ {1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1}
r ↔ {1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0}
s ↔ {1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0}
t ↔ {1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1}
u ↔ {1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1}
v ↔ {1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0}
w ↔ {0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0}
x ↔ {0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0}
y ↔ {0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1}
z ↔ {0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0}
```

Zur Decodierung brauchen wir die Liste Weight2Less aller Vektoren aus $\mathbb{F}_2^{13}$ mit Gewicht höchstens 2 (die Zahl der Fehler, die wir korrigieren wollen) und die Liste Syndromes ihrer Syndrome.

In[75]:=
```
Weight2Less = Join[
    {Array[0 &, 13]},
    Permutations@Join[Array[0 &, 12], {1}],
    Permutations@Join[Array[0 &, 11], {1, 1}]
   ];
Syndromes = Mod[H.#, 2] & /@ Weight2Less
```

Beispiel

Alice hat das 15-te Element der Liste *L* (also den Buchstaben O) gesendet

```
In[133]:= c = L[[15]]
          FromLetterNumber[15]
Out[133]= {0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1}
Out[134]= O
```

Bob hat aber den Vektor

```
In[89]:= x = {0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1};
```

empfangen, der kein Codewort ist. Um zu decodieren, berechnet Bob das Syndrom von *x* und sucht dieses in der Liste Syndromes der Syndrome der möglichen Fehlervektoren

```
Mod[H.x, 2]
Position[Syndromes, %]
Out[98]= {1, 0, 0, 1, 0, 1, 0, 0}
Out[99]= {{55}}
```

Dann ist der 55-te Eintrag der Liste Weight2Less

```
In[100]:= Weight2Less[[55]]
Out[100]= {0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0}
```

der aufgetretene Fehler und

```
In[103]:= Mod[x + Weight2Less[[55]], 2]
          c == %
Out[103]= {0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1}
Out[104]= True
```

das gesendete Codewort.

- Decodieren Sie die folgenden 5 empfangenen Vektoren unter der Voraussetzung, dass jeweils höchstens 2 Fehler aufgetreten sind.
  Was ist die Nachricht?

```
{0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0}
{1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0}
{1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1}
{0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0}
{0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0}
```

- Sei *M* die Menge aller Vektoren aus $\mathbb{F}_2^{13}$ von Gewicht 3. Bestimmen Sie, welche von diesen Vektoren in den Nebenklassen mit Anführer aus der Liste Weight2Less liegen (d.h., für welche $v \in M$ ein $e \in$ Weight2Less existiert mit $v \in e + C$).
- Was passiert bei der Decodierung in der Situation, wenn drei Fehler (der Fehlervektor mit der oben beschriebener Eigenschaft) auftreten? Zeigen Sie dies an einem Beispiel.