

Fehlerkorrektur mit dem Hamming-Code

Hamming-Abstand¹

Beim Entwurf von Codes, die nicht nur eine Fehlererkennung, sondern auch eine Korrektur der Fehler erlauben, spielt der Hamming-Abstand eine wichtige Rolle. Als **Hamming-Abstand** oder Hamming-Distanz bezeichnet man die Anzahl der Stellen, in denen sich zwei Codewörter gleicher Länge unterscheiden. Die Codewörter 0110 und 0000 hätten beispielsweise den Hamming-Abstand 2, da sie sich an 2 Stellen, dem 2. und dem 3. Bit, unterscheiden. Der Hamming-Abstand ist also eine Art Maß dafür, wie unterschiedlich die einzelnen Codewörter eines Codes sind.

Aufgabe 1: Bestimmen Sie für die folgenden Paare von Codewörtern jeweils den Hamming-Abstand

- 111000 und 011000
- 10101010 und 11111111
- 01 und 10

Die Codewörter eines Codes können paarweise unterschiedliche Hamming-Abstände haben. Das Minimum dieser Hamming-Abstände bezeichnet man als **Hamming-Abstand des gesamten Codes**.

Aufgabe 2:

- Zeigen Sie, dass der folgende Code zur Codierung der Ziffern 0 bis 9 den Hamming-Abstand 2 hat.

Ziffer	Codewort	Ziffer	Codewort
0	0000 0000	5	1100 1100
1	0000 1111	6	0000 0011
2	1111 0000	7	0000 1100
3	1111 1111	8	0011 0000
4	0011 0011	9	1100 0000

Abbildung 1: Codewörter für die Ziffern 0 bis 9

- Bestimmen Sie den Hamming-Abstand des Codes, der entsteht, wenn man nur die Codewörter der Ziffern 0 bis 5 zu einem Code zusammenfasst.

Aufgabe 3: Entwerfen Sie für die Farben des Bildes in Abbildung 2 einen Code mit Hamming-Abstand 3. Wie viele Bits benötigen Sie für die Codewörter mindestens?

Aufgabe 4: Begründen Sie, dass für einen Code mit Hamming-Abstand 3 ...

- 1-Bit-Fehler zuverlässig korrigiert werden können.
- 2-Bit-Fehler erkannt, aber nicht zuverlässig korrigiert werden können.
- 3-Bit-Fehler nicht zuverlässig erkannt werden können.

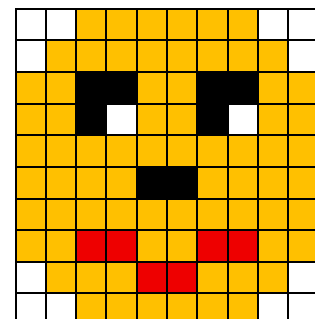


Abbildung 2: Bild mit 4 Farben: Schwarz, Weiß, Rot und Gelb

¹ Hamming-Abstand und Hamming-Code sind nach dem Mathematiker Richard Hamming benannt, der mit seinen Ideen wichtige Grundlagen für das Forschungsfeld der Fehlerkorrektur geschaffen hat (vgl. Manon Bischoff (09.05.2024). Die fabelhafte Welt der Mathematik: Wie Maschinen den Fehlerteufel bekämpfen. <https://www.spektrum.de/kolumne/wie-der-hamming-code-zur-fehlerkorrektur-funktioniert/2214247>

Der (7, 4)-Hamming-Code

Der (7, 4)-Hamming-Code ergänzt eine Bitfolge aus 4 Datenbits nach einem festgelegten Verfahren um 3 Prüfbits, so dass die dadurch entstehenden Bitfolgen aus 7 Bit einen Hamming-Abstand von mindestens 3 haben. Dadurch können im (7, 4)-Hamming-Code 1-Bit-Fehler zuverlässig korrigiert werden.

Erzeugen des (7, 4)-Hamming-Code

Bei den Prüfbits im Hamming-Code handelt es sich um Paritätsbits mit gerader Parität, die jeweils nur für einen Teil der Datenbits erstellt werden. Die Datenbits $d_0 d_1 d_2 d_3$ und Paritätsbits $p_0 p_1 p_2$ werden wie folgt angeordnet: $p_0 p_1 d_0 p_2 d_1 d_2 d_3$

Abbildung 3 zeigt, welche Datenbits bei der Erstellung der Paritätsbits jeweils berücksichtigt werden. Bei der Erstellung des Prüfbits p_i werden jeweils die drei Datenbits berücksichtigt, für die in der entsprechenden Zeile ein Kreuz gesetzt ist.

Abbildung 4 zeigt eine alternative Darstellung in Form eines Venn-Diagramms.

Prüfbit	Datenbits			
	d_0	d_1	d_2	d_3
p_0	x	x	-	x
p_1	x	-	x	x
p_2	-	x	x	x

Abbildung 4: Tabellarische Darstellung der Regeln zur Erstellung der Prüfbits

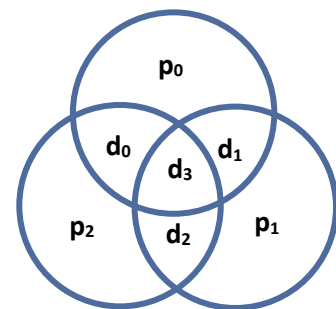


Abbildung 3: Venn-Diagramm

Beispiel

Die Datenbits $d_0 d_1 d_2 d_3 = 1101$ sollen übertragen werden. Vor der Übertragung werden wie folgt drei Prüfbits ergänzt:

$$p_0 = d_0 \oplus d_1 \oplus d_3 = 1 \oplus 1 \oplus 1 = 1$$

$$p_1 = d_0 \oplus d_2 \oplus d_3 = 1 \oplus 0 \oplus 1 = 0$$

$$p_2 = d_1 \oplus d_2 \oplus d_3 = 1 \oplus 0 \oplus 1 = 0$$

Übertragen wird somit die Bitfolge $p_0 p_1 d_0 p_2 d_1 d_2 d_3 = 101101$

Hinweis: Zur Berechnung der Prüfbits wird häufig die logische Operation

xor: \oplus (exklusives oder) verwendet. Die Operation ist so definiert, dass die xor-Verknüpfung der Datenbits gerade dann 1 ergibt, wenn das Prüfbit für eine gerade Parität 1 sein muss, und 0, wenn das Prüfbit für eine gerade Parität 0 sein muss. Abbildung 5 zeigt die Definition der logischen Operation xor.

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Abbildung 5: xor-Verknüpfung

Aufgabe 5: Erstellen Sie den (7, 4)-Hamming-Code für die folgenden Datenbits $d_0 d_1 d_2 d_3$:

- 0011
- 0101

Aufgabe 6: Eine Bitfolge des (7, 4)-Hamming-Code wurde übertragen. Empfangen wurde die Bitfolge 0011000. Zeigen Sie, dass die Bitfolge nicht fehlerfrei übertragen wurde. Erläutern Sie Ihr Vorgehen. Haben Sie schon eine Idee, welches der Bits fehlerhaft übertragen wurde?

Überprüfen eines (7, 4)-Hamming-Code

Um zu überprüfen, ob ein Codewort des (7, 4)-Hamming-Code fehlerfrei übertragen wurde, können die folgenden Rechnungen durchgeführt werden:

$$s_0 = p_0 \oplus d_0 \oplus d_1 \oplus d_3$$

$$s_1 = p_1 \oplus d_0 \oplus d_2 \oplus d_3$$

$$s_2 = p_2 \oplus d_1 \oplus d_2 \oplus d_3$$

Bei einer fehlerfreien Übertragung müssen s_0 , s_1 und s_2 den Wert 0 haben.

Aufgabe 7: Begründen Sie, dass s_0 , s_1 und s_2 den Wert 0 haben müssen, wenn ein Codewort des (7, 4)-Hamming-Code fehlerfrei ist.

Bestimmen der Fehlerposition

Die spezielle Anordnung der Prüfbits ermöglicht im Falle eines 1-Bit-Fehlers die Berechnung der Position des fehlerhaften Bits, so dass dieses korrigiert werden kann: Die Binärzahl $s_2s_1s_0$ zeigt die Position des Fehlers an. Die Positionen werden dabei von links nach rechts beginnend mit 1 gezählt.

Aufgabe 8:

- Wenden Sie das hier beschriebene Verfahren zur Bestimmung der Fehlerposition auf die Bitfolge aus Aufgabe 6 an: 0011000
- Begründen Sie, dass unter der Annahme eines 1-Bit-Fehlers die Binärzahl $s_2s_1s_0$ die Position des fehlerhaften Bits angibt.

Tip: Betrachten Sie dazu die Codierung der Positionen 1 bis 7 als Binärzahl $s_2s_1s_0$ und stellen Sie einen Zusammenhang zur Bildung der Prüfbits p_2 , p_1 und p_0 her.

Position	1	2	3	4	5	6	7
	p_0	p_1	d_0	p_2	d_1	d_2	d_3
s_0	x	-	x	-	x	-	x
s_1	-	x	x	-	-	x	x
s_2	-	-	-	x	x	x	x

- Man findet auch die Definition, dass die Summe der Positionen der fehlerhaften Paritätsbits die Position des fehlerhaften Bits angibt. Erläutern Sie die Gleichwertigkeit zum Vorgehen zur Bestimmung der Position des Fehlers in a).

Aufgabe 9: Untersuchen Sie unter der Annahme, dass höchstens 1-Bit-Fehler auftreten, ob bei der Übertragung der folgenden Bitfolgen gemäß des (7, 4)-Hamming-Codes ein Fehler aufgetreten ist und korrigieren sie ihn ggf.:

- 1011010
- 1100100

Aufgabe 10: Bei der Übertragung der mit dem (7, 4)-Hamming-Code erstellten Bitfolge 0010110 kommt es an den markierten Stellen zu einem 2-Bit-Fehler: 0110100

Zeigen Sie, dass es in diesem Fall durch die Fehlerkorrektur gemäß des (7, 4)-Hamming-Code - Verfahrens zu einer Falschkorrektur kommt.

Erweiterter (7, 4)-Hamming-Code

Aufgabe 11: Um 1-Bit-Fehler von 2-Bit-Fehlern unterscheiden zu können, kann ein weiteres Paritätsbit angehängt werden, dass eine gerade Parität für alle sieben Bit des (7, 4)-Hamming-Code herstellt. Man spricht in diesem Fall von einem erweiterten Hamming-Code.

- Erstellen Sie den erweiterten (7, 4)-Hamming-Code für die Datenbits 0100.
- Zeigen Sie, dass der erweiterte (7, 4)-Hamming-Code einen Hamming-Abstand von 4 hat. Sie dürfen dabei voraussetzen, dass der (7, 4)-Hamming-Code dem Hamming-Abstand 3 hat.
- Begründen Sie, dass mit dem erweiterten (7, 4)-Hamming-Code 1- und 2-Bit-Fehler unterschieden werden können.

Exkurs: Verallgemeinerung des Hamming-Codes

Beim (7, 4)-Hamming-Code ist das Verhältnis von Datenbits zu Prüfbits mit 4:3 relativ schlecht. Nur 4/7 der Bits enthalten Daten. Werden die Prüfbits für größere Datenblöcke erstellt, verbessert sich das Verhältnis deutlich, da die Paritätsbits nur an den Positionen 2^n für $n = 0, 1, 2, \dots$ eingefügt werden müssen.

Ein Hamming-Code mit $2^n - 1$ Bits enthält somit n Prüfbits. Damit können 1-Bit-Fehler in $2^n - n - 1$ Datenbits erkannt und korrigiert werden. Die Tabelle in Abbildung 6 zeigt, welche Datenbits jeweils an der Berechnung der Prüfbits beteiligt sind.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	2^0	2^1		2^2				2^3								2^4															
	p_0	p_1	d_0	p_2	d_1	d_2	d_3	p_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	p_4	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}	d_{21}	d_{22}	d_{23}	d_{24}	d_{25}
p_0	(x)		x		x		x		x		x		x		x		x		x		x		x		x		x		x		x
p_1		(x)	x			x	x			x	x			x	x			x	x			x	x			x	x			x	x
p_2				(x)	x	x	x					x	x	x	x					x	x	x	x					x	x	x	x
p_3								(x)	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x
p_4																(x)	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Abbildung 6: Schema zur Erzeugung der Prüfbits im Hamming-Code

Aufgabe 12:

- Wie viele Datenbits kann ein Hamming-Code mit 4, 5 bzw. 9 Prüfbits enthalten?
- Erläutern Sie die Tabelle in Abbildung 6. Formulieren Sie eine allgemeine Regel, welche Datenbits in die Berechnung des Prüfbits an Position n eingehen.

Aufgabe 13:

- Erstellen Sie den (15, 11)-Hamming-Code für die Bitfolge 1011 0001 111. Markieren Sie dazu in der folgenden Vorlage zunächst die Datenbits, die das jeweilige Prüfbit beeinflussen.

p_0 : __ 1 __ 0 1 1 __ 0 0 0 1 1 1 1 \rightarrow $p_0 =$ ____

p_1 : __ 1 __ 0 1 1 __ 0 0 0 1 1 1 1 \rightarrow $p_1 =$ ____

p_2 : __ 1 __ 0 1 1 __ 0 0 0 1 1 1 1 \rightarrow $p_2 =$ ____

p_3 : __ 1 __ 0 1 1 __ 0 0 0 1 1 1 1 \rightarrow $p_3 =$ ____

(15,11)-Hamming-Code: __ __ 1 __ 0 1 1 __ 0 0 0 1 1 1 1

- Erstellen Sie den (31, 26)-Hamming-Code für die Bitfolge 1101 1111 0000 1001 1010 1100 10
- Zeigen Sie, dass die Bitfolge 0101 0111 1111 011, die mit dem (15, 11)-Hamming-Code erstellt wurde, nicht korrekt übertragen wurde und korrigieren Sie den Fehler unter der Annahme, dass nur 1 Bit fehlerhaft ist.

Aufgabe 14:

- Nehmen Sie Stellung zu der Aussage „Kompression und Fehlerkorrektur stehen im Widerspruch zueinander“.
- Diskutieren Sie, in welcher Reihenfolge eine Codierung zur Kompression und zur Fehlererkennung durchgeführt werden sollten, wenn beide Codierungen miteinander kombiniert werden sollen.

Implementierung

Aufgabe 15: Erstellen Sie ein Programm, das ...

- den (7, 4)-Hamming-Code zu einer Zeichenkette aus 4 Bit erstellt.
- ein Codewort mit 7 Bit nach dem (7, 4)-Hamming-Code-Verfahren auf Korrektheit überprüft und ggf. unter der Annahme eines 1-Bit-Fehlers korrigiert.
- das Übertragen und Korrigieren von 4 Datenbits mithilfe des (7, 4)-Hamming-Codes simuliert.

Dazu können folgende Schritte durchgeführt werden:

- [1] Erzeugen einer zufälligen Folge aus 4 Bit
- [2] Erweitern der 4 Bit zum (7, 4)-Hamming-Code
- [3] Erzeugen eines Fehlers an einer zufälligen Position
- [4] Überprüfen und korrigieren des fehlerhaften Codeworts
- [5] Auslesen der Datenbits aus dem Hamming-Code

Aufgabe 16:

- Begründen Sie, dass der (7, 4)-Hamming-Code genau 16 verschiedene Codewörter umfasst.
- Erstellen Sie ein Programm, das eine Tabelle mit allen 16 Codewörtern des (7, 4)-Hamming-Codes und ihren paarweisen Hamming-Abständen erzeugt. Abbildung 7 zeigt den Anfang einer solchen Tabelle.

Hamming	0000000	1101001	...
0000000	0	4	...
1101001	4	0	...
...

Abbildung 7: Systematische Auflistung aller (7, 4)-Hamming-Codes und ihre paarweisen Hamming-Abstände

Fehlerkorrigierende Codes im Alltag

Neben Barcodes findet man auf vielen Produkten, Weberflyern usw. QR-Codes.

Aufgabe 17: Sammeln Sie Beispiele für Informationen, die in Form eines QR-Codes dargestellt werden. Warum könnte eine Darstellung als QR-Code in diesen Fällen sinnvoll sein?

Aufgabe 18: Abbildung 8 zeigt drei QR-Codes, die teilweise beschädigt sind.

a) Testen Sie, welche der QR-Codes noch lesbar sind.

Hinweis: Bei vielen Smartphones können QR-Codes mit der Kamera-App decodiert werden.

b) Versuchen Sie, Ihre Beobachtungen mithilfe Ihres Wissens über Fehlererkennung und -korrektur zu erklären. Stellen Sie eine Vermutung auf, wie QR-Codes aufgebaut sind.



Abbildung 8: beschädigte QR-Codes

Richard Hamming war einer der ersten, der einen Code zur Fehlerkorrektur entwickelte. Er veröffentlichte seinen Code bereits 1950². Der nach ihm benannte Hamming-Code kommt heute noch zur Fehlerkorrektur in Speichermedien oder in der Satellitenkommunikation zum Einsatz. Auf der Grundlage von Hamming's Forschung und Ideen wurden später weitere fehlerkorrigierende Codes entwickelt. In QR-Codes wird z. B. ein Reed-Solomon Code verwendet, mit dem mehrere Fehler in einem Datenblock erkannt und korrigiert werden können.

Aufgabe 19: Begründen Sie, dass der Hamming-Code besser für die Fehlerkorrektur bei der Nachrichtenübertragung als für die Fehlerkorrektur in einem QR-Code geeignet ist.

Dieses Werk ist lizenziert unter einer [Creative Commons Namensnennung - Nicht-kommerziell - Weitergabe unter gleichen Bedingungen 4.0 International Lizenz](#). Von der Lizenz ausgenommen ist das InfSII-Logo.

² Richard W. Hamming: *Error Detection and Error Correction Codes*. In: *The Bell System Technical Journal*, Vol. XXIX 2, 1950, Seite 147–160.