Grundkurs Codierung Auflage 5

Lösungsvorschläge zu den Fragen in den Unterkapiteln „Was blieb?“

Stand 20.08.2025

Zu Unterkapitel 1.6 Seite 26

Zu Frage 1:

a) 1-, 2-und 3-Bitfehler-Erkennung

b) 1-Bitfehler-Korrektur oder 2-Bitfehler-Erkennung. **Aber Vorsicht:** Man muss sich beim Codierungsverfahren entscheiden, ob man a) oder b) will, beides zugleich geht nicht, warum?

Zu Frage 2:

a) Fehlerkorrigierbarkeit erhöht die Redundanz einer Information, da dieser Information die aus den Infobits berechneten Prüfbits hinzugefügt werden.

c) Verschlüsselung ändert die Redundanz in der Information im Allgemeinen nicht, sondern "verwirbelt" nur die Infobits.

a) Datenkompression entfernt (überflüssige) Redundanz.

Zu Frage 3:

Gemäß S. 13 ist der Informationsgehalt H (= Entropie) eines einzelnen Zeichens

$$H = \mathrm{ld}\frac{1}{p}, \qquad \mathrm{ld} = \text{Zweier-Logarithmus} \quad .$$

Für p = 0.125 = 0.5^3 beträgt H = 3: Davon zu unterscheiden ist die mittlere Entropie eines Zeichen-Alphabets. Unter der Annahme, dass das Alphabet im vorliegenden Fall nur aus zwei Zeichen bestünde, hätte das zweite Zeichen eine Eintrittswahrscheinlichkeit von 1 – p = 0.875. Die mittlere Entropie des Alphabets wäre nach S. 13 also

$$H = p\cdot\mathrm{ld}\frac{1}{p} + (1-p)\cdot\mathrm{ld}\frac{1}{1-p} = 0.543564$$

.

Sie ist geringer als für gleich wahrscheinliche Zeichen (wie groß wäre diese?).

Zu Frage 4:

a) Überlegen Sie sich zunächst, welche Programmteil hierfür erforderlich oder sind. Siehe z.B. Seiten 22/23:

- Erzeugung der 2^6 = 64 Wörter
- Funktion zur Abstandsberechnung zweier Wörter (=Skalarprodukt)
- Abstandsberechnung für alle möglichen Paarungen der 64 Wörter
- Darstellung in einer 64· 64 – Matrix
- Bestimmung der Wörter mit Abstand 3 zum ersten Wort
- Streichen der Spalten und Zeilen mit Abstand < 3 zum ersten Wort
- Bestimmung der Wörter mit Abstand 3 zum zweiten Wort
- ... und so weiter, bis man beim letzten verbliebenen Wort angekommen ist.
- Entsprechend verfährt man für den Abstand 4.

b) Erstellung des Programms (C++, Matlab, Mathematica, ….)

Zu Frage 5:

Für die Lösung sind die Angaben im Unterkapitel 3.5.6, S. 83/84, nützlich. Es ist ein bestimmtes Integral in den Grenzen von - ∞ bis + ∞ zu lösen:

$$H(x) = -\int_{-\infty}^{+\infty} p(x) \cdot \mathrm{ld}\, p(x)\, dx = \mathrm{ld}\sqrt{2 \cdot \pi \cdot e \cdot \sigma^2} \quad .$$

Schauen Sie in einer ausführlichen Formelsammlung nach, z. B. in Bronstein Semendjajew, “Taschenbuch der Mathematik”

Zu Frage 6:

Der Morsecode nutzt die unterschiedlichen Zeichenhäufigkeiten des Alphabets einer natürlichen Sprache. Häufig vorkommende Zeichen der deutschen oder auch der englischen Sprache (s. S. 254, Bild 4.14) wie der “Zwischenraum”, das “e”, das “n” usw. erhalten kurze Codewörter, seltenere Zeichen wie das “j” oder das “q” längere. Im Mittel ergibt sich also eine Einsparung bei der Summe zu übertragender Symbole (beim Morsealphabet sind es die Punkte und Striche).

Ähnlich arbeitet auch der Huffman-Code, s. Unterkapitel 6.1.2, S. 325 ff. Allerdings gibt es einen wesentlichen Unterschied: Beim Morsecode müssen die Codewörter durch eine deutliche Pause (bei akustischer Übermittlung) oder einen klaren Zwischenraum (bei geschriebenen Punkten und Strichen) getrennt werden, was zusätzlichen Aufwand erzeugt. Dies ist beim Huffman-Code nicht erforderlich (warum?).

Zu Frage 7:

Ja, denn durch das Anhängen des Prüfbits werden die Infobits nicht verändert. Im Übrigen ist es für systematische Codes zwar nicht notwendig, dass die Infostellen zusammen bleiben (also z. B. alle “vorn” im Codewort stehen), die Decodierung läuft dann aber einfacher ab, weil zusätzliche Arbeitsschritte zu ihrer Gruppierung entfallen.

Zu Frage 8:

Für einen Kanalwiderstand R (Wellenwiderstand Z, dieser ist oft näherungsweise reell) ist der Störabstand

$$\mathrm{SNR} = \frac{\text{Nutzsignalleistung}}{\text{Störsignalleistung}} = \frac{N_{nutz}}{N_{stör}} = \frac{\frac{U_{nutz}^2}{R}}{\frac{U_{stör}^2}{R}} = \frac{U_{nutz}^2}{U_{stör}^2} = \frac{2.5^2}{1.3^2} = 3.7 \quad \text{oder} \quad \mathrm{SNR} = 10 \cdot \log_{10}(3.7) = 5.7 \ \mathrm{db}$$

Zu Frage 9:

a) Jedes Klartextzeichen oder Klartextwort wird durch ein eigenes, zufällig gewähltes Schlüsselzeichen oder Schlüsselwort verschlüsselt.

b) Selbst dann, wenn ein Angreifer alle möglichen Schlüssel durchprobieren würde (“brute force”-Angriff), bekäme er – zumindest bei sehr langen Geheimtexten - eventuell mehrere gleichwertige, aber inhaltlich unterschiedlichster Klartextergebnisse.

c) Die Übertragung der Schlüssel zwischen den berechtigten Partnern war bei symmetrischer Verschlüsselung lange Zeit eine große Schwachstelle. Durch die Erfindung der asymmetrischen Verfahren (z.B. RSA) wurde diese - zumindest vorübergehend – beseitigt.

d) Für die Erzeugung der zufälligen Schlüssel muss sichergestellt werden, dass diese wirklich “ausreichend” zufällig sind, was besondere Aufmerksamkeit bei der Wahl geeigneter Algorithmen erfordert.

Zu Frage 10:

Jedes Codewort besitzt eine Abfolge von Bits, die eindeutig im Blatt des zugeordneten Zeichens endet (s. auch den Codebaum im Unterkapitel 6.1.2, S. 326). Das Folgebit gehört daher bereits zum nächsten Codewort (“**Prefix-Eigenschaft**” der Codewörter). Zusatzfrage: Ist dies eine spezielle Eigenschaft des Huffman-Codes? Man betrachte hierzu einmal den ASCII-Code. Liegen hier die Verhältnisse grundsätzlich anders? Wenn nein, warum ist der ASCII-Code dann trotzdem weniger platzsparend? Und noch eine Frage: Wie robust verhält sich der Huffman-Code gegen Übertragungsfehler? Ist nach einem Fehler das nächste Zeichen sofort wieder zuordenbar?

Zu Frage 11:

Ein Baum des Morse-Codes zeigt, dass nicht allen Codewörtern eindeutig Zeichen als Blatt zugeordnet sind. Z. B. ist

5 = ...
h = ...
s = ...
i = ..
e = .

Trifft nun beim Empfänger eine Folge “.....” ein, so weiss er nicht, ob es sich um “5”, oder um “eh” oder um “eis” oder eine andere mögliche Kombination handelt (man vergleiche mit dem Huffman-Code). Mit Einfügen von Pausen zwischen den Codewörtern wird die Eindeutigkeit hergestellt, allerdings auf Kosten der Kompressionsrate. Hinweis für die Amateurfunker: Die Pausen haben auch einen Betriebsvorteil, den man sieht, wenn man statt des Morse-Codes den Huffman-Code verwenden würde. Dann müsste man die Abfolge der eintreffenden Bits “im Kopf” in Echtzeit auf den Codebaum abbilden. Das wäre zwar machbar, beim ersten Fehler hätte man aber bereits die Synchronisation verloren.

Zu Frage 12:

Nein, es gibt keine Definition dafür.

Zu Frage 13:

Da es eine Vielzahl von gleichwertigen Generatormatrizen gibt, die durch Permutationstransformationen auseinander gebildet werden können, erzeugen auch viele von ihnen nichtsystematische Codewörter. Die in der Frage genannte Tabelle 1.2 bekam in der Auflage einen anderen Inhalt. Die alte Tabelle 1.2 ist nicht mehr enthalten. Überlegen Sie sich bitte selbst ein Beispiel.