Scientific Computing

Universität Salzburg

VP

Multimediakompression

SS 2003

Beispiele

1. Gegeben ist die Symbolfolge *ccacccbc*.

   (a) Berechne die Wahrscheinlichkeiten $p(a), p(b), p(c)$ aufgrund ihrer relativen Häufigkeiten.

   (b) Berechne den Informationsgehalt der Folge.

   (c) Entwickle den Huffman-Code und gib die Kodierung an.

   (d) Führe die arithmetische Kodierung durch. (aufwändig aber lehrreich!)

2. Das Signal $f(t) = 3\cos(\omega t + \frac{\pi}{16})$ wird an den Stellen $t_n = \frac{n\pi}{8\omega}$ gesamplet (gerundet). Das Ergebnis $g(n) = \mathrm{round}(f(t_n))$ wird als Symbolquelle betrachtet.

   (a) Berechne die Entropie der Quelle unter der Annahme, dass die Samples (Werte) voneinander unabhängig sind. Welchen Informationsgehalt haben 240 Symbole?

   (b) Fasse je zwei Symbole zusammen und betrachte die entstehenden Symbole als unabhängig. Berechne die Entropie und den Informationsgehalt von 240 (jetzt 120) Symbolen.

   (c) Dasselbe für je drei zusammengefasste Symbole.

   (d) Berechne nun das Differenzsignal $g(n+1) - g(n)$. Entropie und Informationsgehalt für 240 Symbole.

   (e) Das Signal wird nun durch einen einfachen linearen Prediktor approximiert, für den gilt: $\bar{g}(n+1) - g(n) = g(n) - g(n-1)$. $\bar{g}(n)$ ist die Approximation. Berechne den mittleren quadratischen Fehler (MSE) der Approximation sowie Entropie und Informationsgehalt des Differenzsignals (240 Symbole).

   (f) Berechne nun die optimalen Koeffizienten a_1 und a_2 des linearen Prediktors $\bar{q}(n) = a_1 q(n-1) + a_2 q(n-2)$. Wie oben MSE, Entropie und I.-gehalt.

3. Gegeben ist folgende Symbolfolge: 10120012300012340000…1234567800000000.

   (a) Berechne Entropie und Informationsgehalt unter der Annahme, dass die Symbole unabhängig voneinander sind.

   (b) Die Folge wird Lauflängen-codiert mit den Symbolen 1, 2, …8, c_1, c_2, …, c_n. Die c_i werden anstatt einer Folge von i 0-en eingefügt. Berechne Entropie und Informationsgehalt der sich ergebenden Symbolfolge für (1) $n = 8$ und (2) $n = 4$.

   (c) Das gleiche mit dem zusätzlichen Symbol 0. Dieses fungiert als Kontext-Switch: Immer nach einer 0 folgt ein c_i mit obiger Bedeutung.

   (d) Entwickle jeweils den Huffman-Code für (a), (b.1), (b.2), (c.1) und (c.2) und ermittle die gesamte und mittlere Codelänge.

4. Gegeben ist ein Code, der alle natürlichen Zahlen (0, 1, 2, …) darstellen kann. Dabei werden, um die Zahl n zu codieren, einfach n 0-en gefolgt von einer 1 auf den Bitstream geschrieben. Also 5 wird z.B. zu 000001. Die Wahrscheinlichkeit, dass die Zahl n auftritt, ist $p(n) = 2^{-(n+1)}$. Berechne die Entropie und die mittlere Codelänge dieses Codes. (Bem.: es entsteht eine konvergierende Reihe. Wenn du den Wert der Reihe nicht analytisch ermitteln kannst, berechne die Summe der ersten paar Reihenglieder und schätze das Ergebnis.)

5. Die Symbolfolge AABABCABCDABCDEABCDEF soll kodiert werden.

   (a) Berechne den Informationsgehalt, entwickle einen Huffman-Code und ermittle die Anzahl der Bits für die Huffman-Kodierung der Symbolfolge.

   (b) Benutze Dictionary Coding nach dem LZ77-Schema. Es sind nur Matches der Länge 2 und 3 zugelassen. Kodiere Position und Länge der Matches als separate Symbole. Berechne Informationsgehalt der resultierenden Symbolfolge, entwickle den Huffman-Code und ermittle die Anzahl der Bits für die Huffman-Kodierung. Achte dabei auf mögliche Kontexttrennung.

   (c) Wie (b). Es sind jedoch Matches beliebiger Länge zulässig. Fasse außerdem Position und Länge der Matches zu einem Symbol zusammen.

   (d) Führe die Burrows-Wheeler-Transformation auf der Symbolfolge aus und verzichte dabei auf die Endekennung. Hint: Um den Aufwand in Grenzen zu halten kann man die Zeilen der Matrix auf die ersten paar Zeichen plus das letzte beschränken (und vielleicht gleich geordnet anschreiben). Kodiere das Ergebnis mittels Lauflängenkodierung (Methode: Modusselektion). Berechne Informationsgehalt und Huffmankodierung.

6. Gegeben ist das Signal $f(t) = \sin(\omega t)$. Das Signal wird nach μ-law non-uniform mit einem Dead-zone-Quantizer quantisiert. Der Quantisierungsfaktor ist $q = 4$. Berechne allgemein (in Abhängigkeit von μ) für jedes Bin die Wahrscheinlichkeit, dass $f(t)$ in das Bin fällt. Der Quantisierungsfehler wird für jedes Bin mit $(o - u)^2/4$ abgeschätzt (o ist die obere, u die untere Bingrenze). Finde durch Probieren jene ganze Zahl μ, die den gesamten Fehler minimiert.

7. Gegeben sind folgende Elemente des $\mathbf{R}^2$: $\{A(\cos(2\pi n/5), \sin(2\pi n/5))|n = 0 \ldots 4, A \in \{8, 4\}\}$. Führe den LBG-Algorithmus für 4 Quantisierungsvektoren durch. Verwende dabei ein "Auseinanderrücken" im 45-Grad-Winkel. (Aufzeichnen mit Lineal und Entfernungen messen ist erlaubt.) Berechne den Quantisierungsfehler.

8. Ein 2-D Array mit 8×8 Elementen soll mittels Quadtree- und Bitplane-Kodierung kodiert werden. Die Elemente sind definiert durch $f(i, j) = 15 + 15 \sin(2\pi i/8) \sin(2\pi j/8)$ für $0 \leq i, j \leq 7$. Kodiere eine Viertelung ("P") mit 1, einen 1-Bereich mit 01 und einen 0-Bereich mit 00. Kodiere 3 Bitplanes. Zähle nach jedem Bitplane die Anzahl der geschriebenen Bits, berechne die Rekonstruktion und die PSNR.