Letztes Mal

Huffman-Algorithmus: Finde optimalen Präfixcode für gegebene Häufigkeiten

- gieriger Algorithmus: baue Baum von unten, vereinige Bäume mit geringsten Gesamthäufigkeiten

Satz Der Huffman-Algorithmus liefert einen optimalen präfixfreien Code für gegebene Häufigkeiten.

Lemma (Struktur einer optimalen Lösung):

Seien σ und τ Zeichen mit kleinstmöglichen Häufigkeiten. Dann existiert ein optimaler präfixfreier Code, in dem σ und τ Geschwister sind und maximale Tiefe haben.

Beweis durch Austauschargument.

Beweis des Satzes:

Induktion nach $k = |\Sigma|$.

Induktionsanfang: $k = 1$ ✓

$k = 2$ 0,1 ist optimaler Code

Induktionsschritt:

Annahme HK ist optimal für alle Alphabete der Größe $k-1$ und alle Häufigkeiten

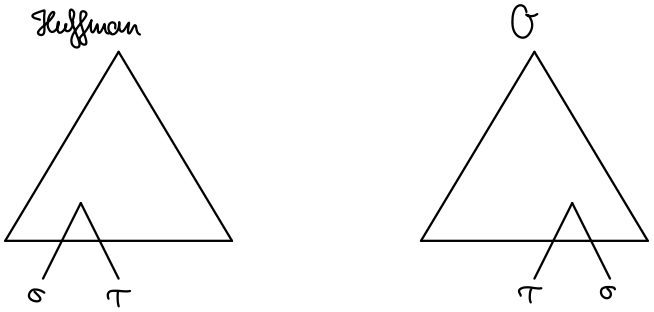
z.z. HK ist optimal für alle Alphabete der Größe k und alle Häufigkeiten

Nimm an: Es gibt Häufigkeiten $h_{\sigma_1}, \dots, h_{\sigma_k}$ für die HK nicht optimal ist.

Seien σ und τ die Symbole, die Huffman-Algorithmus zuerst vereinigt

Nach L1 existiert ein optimaler Code O, in dem σ und τ Geschwister sind.

Also

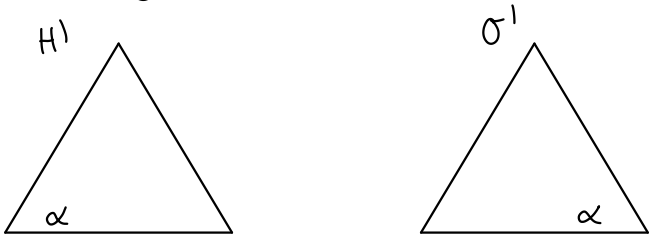


und $\sum_{i=1}^{k} |H(\sigma_i)|\, h_{\sigma_i} > \sum_{i=1}^{k} |O(\sigma_i)|\, h_{\sigma_i}$

Sei α neues Symbol mit $h_\alpha = h_\sigma + h_\tau$.

Definiere Alphabet $\Sigma' = \Sigma \setminus \{\sigma, \tau\} \cup \{\alpha\}$.

Betrachte Kodes



H' und O' sind Präfixcodes für Σ'

H' ist ein Huffmankode für Σ', also nach Induktionsannahme optimal.

Beh O' ist besser als H'.

Betrachte dazu

$$\sum_{i=1}^{k-1} |H'(\sigma_i)|\, h'_{\sigma_i} - \sum_{i=1}^{k-1} |O'(\sigma_i)|\, h'_{\sigma_i} = \sum_{i=1}^{k} |H(\sigma_i)|\, h_{\sigma_i} - \sum_{i=1}^{k} |O(\sigma_i)|\, h_{\sigma_i} < 0$$

Denn $|H(\sigma)|\, h_\sigma + |H(\tau)|\, h_\tau - |O(\sigma)|\, h_\sigma - |O(\tau)|\, h_\tau$

$= (|H'(\alpha)|+1)\, h_\sigma + (|H'(\alpha)|+1)\, h_\tau - (|O'(\alpha)|+1)\, h_\sigma - (|O'(\alpha)|+1)\, h_\tau$

$= |H'(\alpha)|\, h'_\alpha - |O'(\alpha)|\, h'_\alpha$

Dies widerspricht der I. A. Also muss H optimal gewesen sein.

Denn es ist $|H(\sigma)| = |H(\tau)| = |H'(\alpha)|+1$

und $|O(\sigma)| = |O(\tau)| = |O'(\alpha)|+1$ □

Dateikompression

- Huffmankodes sind optimal, wenn nur zeichenweise kodiert wird.
- Andere Methoden, die Längeneinheiten kodieren:
  - Run-Length-Encode RLE & PCV

    1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 → 6 7 3
  - Lempel-Ziv-Welch & Varianten:
    - Erkenne wiederholende Wörter / Muster
    - Compress, GIF, etc.
    - patentiert (ausgelaufen?)
- verlustbehaftete Verfahren: DECODE ≠ ORIGINAL

  Bei Bildern, Videos, Ton dürfen Informationen verloren gehen, speichere nur „wichtige" Daten:
  - JPEG, MP3, MP4, etc.
    - Fast-Fourier-Transform
    - Discrete-Cosine-Transform
    - Wavelet-Transform

Neues Problem: Wie ähnlich sind zwei Zeichenketten?

z.B. Unix-Befehle diff, fc, cmp

Formal Finde längste gemeinsame Teilfolge: LGT/LCS

$S = s_1 s_2 \dots s_n$

$t = t_1 t_2 \dots t_\ell$

Gemeinsame Teilfolge $\begin{matrix} s_{i_1} & s_{i_2} & \dots & s_{i_z} \\ \| & \| & & \| \\ t_{j_1} & t_{j_2} & & t_{j_z} \end{matrix}$ $i_1 < i_2 < \dots < i_z$ $j_1 < j_2 < \dots < j_z$

Wie findet man LGT mit maximaler Länge?

z.B. WEIHNACHTEN

NEUJAHR

Gemeinsame TF: NAH

Wie findet man LCS?

Rekursion von hinten! + Dynamisches Programmieren.