Greedy (gierig) Algorithmen

gebrochenes Rucksackproblem

gegeben: Gegenstände $E = \{(v_i, w_i), i = 1..n\}$

gesucht Anteile $0 \le q_i \le 1$ s.d.

$\sum_{i=1}^{n} q_i v_i$ maximal

unter $\sum q_i w_i \le W$

sortiere E so dass $v_i/w_i \ge v_{i+1}/w_{i+1}$ $\quad i = 1..n-1$

sei $j = \max\{0 \le k \le n \mid \sum w_i \le W\}$

$\Rightarrow$ $q_i = 1 \quad \forall i \le j$

$q_{j+1} = \frac{W - \sum w_i}{w_{j+1}}$

$q_i = 0 \quad \forall i > j+1$

$\Rightarrow \Theta(n \log n)$ (sortieren + Berechnung)

Korrektheit:

$(r_i)_{i=1..n}$ eine optimale Lösung, $(r_i) \neq (q_i)$

Rucksack immer voll: $\sum r_i w_i = \sum q_i w_i = W$

Sei $k = \min\{1 \le i \le n : r_i \neq q_i\} \Rightarrow q_k > r_k$ (nach Konstruktion)

$X := q_k - r_k > 0$ [r verteilt mehr Gewicht auf $\{k+1..n\}$ als q]

Konstruktion neue Lösung (r_i'): $r_i' = r_i = q_i \quad \forall i < k$

$r_k' = q_k$ [r' so wie q für $[1..k]$]

Entferne Gewicht $\sum_{i=k+1}^{n} \delta_i = x \cdot w_k = (q_k - r_k) w_k$ von Gegenständen $\{k+1, \dots, n\}$. Möglich, denn $\sum r_i w_i = \sum q_i w_i$

$$\sum_{i=k}^{n} r_i' v_i = \overbrace{r_k v_k + x w_k \frac{v_k}{w_k}}^{q_k v_k} + \sum_{i=k+1}^{n} (r_i w_i - \delta_i) \frac{v_i}{w_i}$$

$$\geq \text{“} \qquad \text{“} + \sum_{i=k+1}^{n} r_i w_i \frac{v_i}{w_i} - \delta_i \frac{v_k}{w_k} \quad \text{(sortiert)}$$

$$= r_k v_k + x w_k \frac{v_k}{w_k} - x \cdot w_k \frac{v_k}{w_k} + \sum_{i=k+1}^{n} r_i w_i \frac{v_i}{w_i}$$

$$= \sum_{i=k}^{n} r_i v_i \quad \Rightarrow r_i' \text{ auch optimal} \Rightarrow \text{Induktion} \quad \square$$

Huffman Codierung

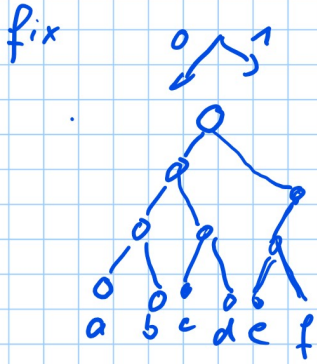
gegeben: langer Text aus Zeichen $c \in C$ mit Frequenzen $f(c)$

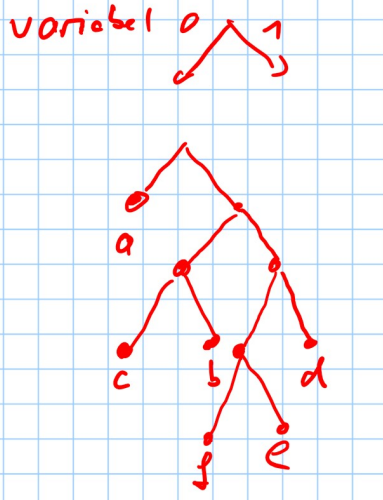
Beispiel

	a	b	c	d	e	f	$\sum f(c) \cdot \#\text{bits}(c)$
f	45k	13k	12k	16k	9k	5k	
fix	000	001	010	011	100	101	300k
variabel	0	101	100	111	1101	1100	224k

gesucht: binärer Zeichencode aus Codewörtern

Codebäume





001 000 011

b a d

101 0 111

b a d

Präfix codes

(a) kein Codewort beginnt mit anderem Codewort

(b) Präfixcodes können optimale Kompression erreichen (ohne Beweis)

(c) Codierung: Verkettung Codewörter ohne Zwischenzeichen
[Unterschied zum Morsen]

affe → 0 1100 1100 1101

(d) Decodierung einfach (Präfixcode!)

0 1100 1100 1101 → affe

Codebäume Eigenschaften

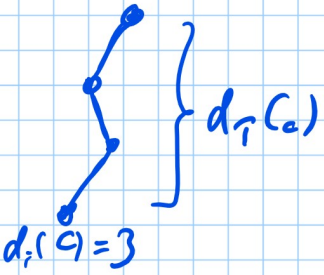
Optimale Codierung

↪ vollständiger binärer Baum

def: Codewörter C

$f(c)$ Häufigkeit von c

$d_T(c)$ Tiefe von c in Baum T



Kosten: $B(T) = \sum_{c \in C} f(c) \cdot d_T(c)$

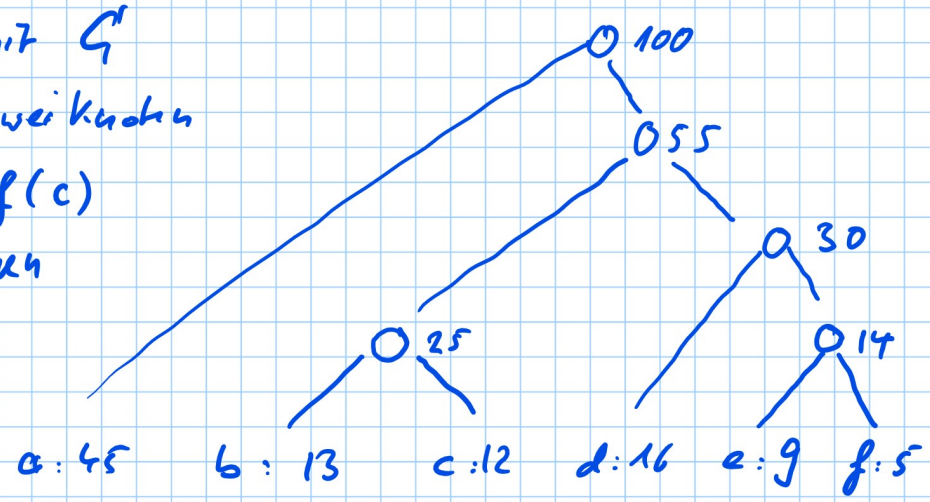
(Anzahl Bits des Files)

optimaler Baum: $B(T)$ minimal

Konstruktion eines optimalen Baumes

(a) Starte mit C

(b) ersetze zwei Knoten mit min. $f(c)$ durch neuen Vaterknoten



Algorithmus

$|n| \leftarrow |C|$

$Q \leftarrow C$ Q MinHeap $\Theta(n)$

for $i \leftarrow 1..n$ $n\times$

    new node z

    $z.left \leftarrow Q.extractMin$ $\Theta(\log n)$

    $z.right \leftarrow$ " " $\Theta(\log n)$

    $z.freq \leftarrow f(z.left) + f(z.right)$ $\Theta(1)$

    $Q \leftarrow Q \cup \{z\}$ $\Theta(\log n)$

return $Q.extractMin()$;

$\Theta(n \log n)$

Korrektheit

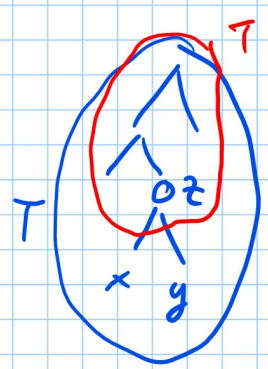
$$B(T) = \sum_{c \in C} f(c)\, d_T(c)$$

x, y: $f(x) \le f(t)$

$f(y) \le f(t) \quad \forall t \in C \setminus \{x, y\}$

$T' \subset C'$: $C' = C - \{x, y\} + \{z\}$

$f(z) = f(x) + f(y)$



Zeige: T' optimal für $C' \Rightarrow T$ optimal für C

Beweis:

Beobachtung:

$$\begin{aligned} & f(x) \cdot d_T(x) + f(y) \cdot d_T(y) \\ = & f(x) \cdot (d_{T'}(z) + 1) + f(y) \cdot (d_{T'}(z) + 1) \\ = & d_{T'}(z) \cdot f(z) + f(x) + f(y) \end{aligned}$$

$$\Rightarrow B(T') = B(T) - f(x) - f(y)$$

Annahme: T nicht optimal $\Rightarrow \exists T'', B(T'') < B(T)$

x, y Brüder in $T'' \Rightarrow$

sei T''' Baum T'' mit $(x, y) \to (z)$

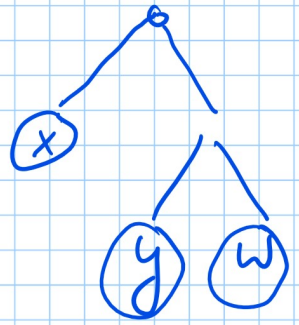
$$\Rightarrow B(T''') = B(T'') - f(x) - f(y)$$

$$< B(T) - f(x) - f(y) - B(T')$$

$\Rightarrow T'$ nicht optimal ↯

x, y Brücke in T''

(a)



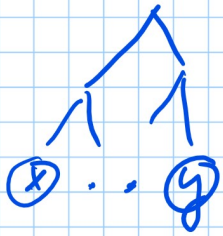
Tausch $x \leftrightarrow w : T'' \to \tilde{T}''$

$(f(x) \leq f(w))$

$\to \underbrace{d_T(x)}_{< d_T(w)} \cdot f(x) + d_T(w) \cdot f(w)$

$\to B(\tilde{T}) < B(T'')$ ↯

(b)



Tausche x od. $y : T'' \to \tilde{T}''$

$\to B(T'')$ bleibt gleich

ersetze T'' durch $\tilde{T}''$