Huffman Code

Albert-Einstein-Schule Ettlingen — 23. Juni, 2019

1 Einleitung

Um Daten zu komprimieren, kann der Huffman Code genutzt werden. Die Huffman Codierung macht sich die Tatsache zunutze, dass Buchstaben unterschiedlich oft in Texten vorkommen. Die Länge der codierten Wörter im Huffman Code hängt dabei mit dessen Informationsgehalt zusammen. Der Huffman Code benutzt dabei einen binären Baum und wird von den Blättern zur Wurzel kodiert. Der Huffman Code ist dabei präfixfrei. Dies heißt, dass die Codewörter keine Trennung brauchen.

2 Algorithmus

[3]

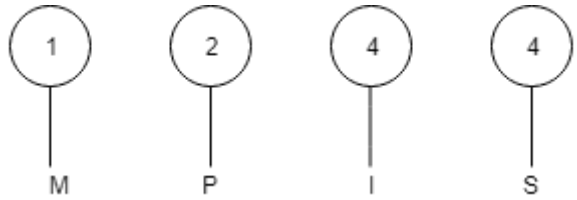
2.1 Codierung

Bevor man den Huffman Code codieren kann, muss erst mal ein Zeichensatz C festgelegt werden. Jedem Zeichen c ∈ C wird eine bestimmte Häufigkeit f(c) zugewiesen. Daraus ergibt sich eine Prioritätenkette Q.

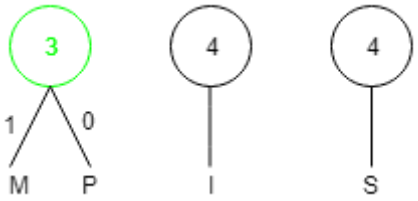
Hier ein Beispiel mit dem Wort MISSISSIPPI

⇒ C = { M,I,S,P } mit Häufigkeit ergibt sich Q:

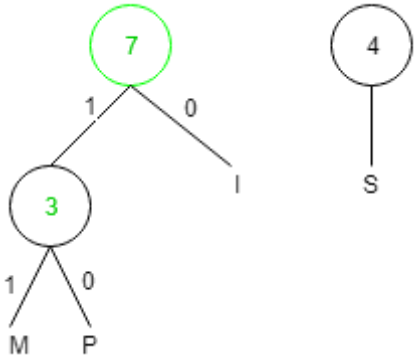
M	P	I	S
1	2	4	4



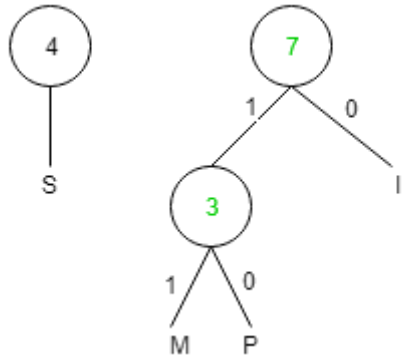
Jetzt werden die beiden Zeichen c mit der geringsten Häufigkeit f(c) mit einer Astgabelung zusammengeführt. Die beiden Häufigkeiten werden addiert und in die Astgabelung geschrieben.



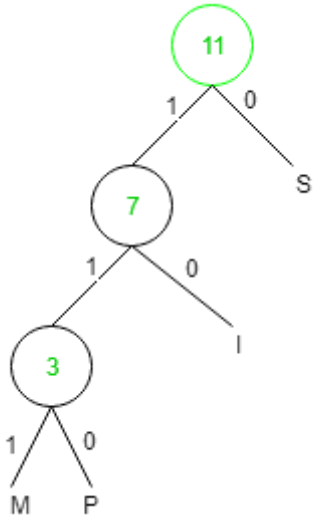
Dieses Vorgehen wird fortgesetzt und es werden wieder die beiden Äste mit den geringsten Häufigkeiten zusammengeführt. Die Astgabelung wird dabei behandelt wie ein Einzelnes Zeichen.



Es muss immer darauf geachtet werden die Zahlen in den Astgabelungen geordnet zu halten. Deshalb wird die Prioritätenkette neu geordnet.



Nach der Neuordnung können wieder die beiden kleinsten Äste zusammengeführt werden und der Huffmanbaum ist fertig.



Jetzt kann aus dem Huffmanbaum die Codierung für jedes Zeichen abgelesen werden.

S	I	P	M
0	10	110	111

3 Decodierung

Die Decodierung des Huffman Codes ist sehr einfach, da es sich beim Huffman Code um einen präfixfreien Code handelt. Das bedeutet man benötigt keine Trennung in der Codierung zwischen den Zeichen. Bei der Decodierung folgt man dem Baum von oben nach unten bis man das Zeichen hat.

4 Anwendungsfälle

[1]

Der Huffman Code wird in zahlreichen Bereichen zur Komprimierung von Daten genutzt. Dies ist beispielsweise bei verlustbehafteter Kompression, wie bei JPEG der Fall. Dabei wird weniger wert auf Bildqualität, sondern mehr darauf die Dateigröße gering zu halten. Ein Beispiel hierfür ist das Internet. Dort ist es aufgrund der Übertragungsgeschwindigkeiten des Netzwerks von Vorteil die Bilddaten zu reduzieren, was geringere Ladezeiten als Folge hat.

5 Optimalität

[5]

Beim Anwenden des Huffman Codes, wird jeder Buchstabe oder Zeichen durch einen spezifischen Code ersetzt. Ein optimaler Code, würde dabei die Codewortlänge mindestens um die durchschnittliche Codewortlänge minimieren. Mathematisch kann man die mittlere Codewortlänge durch eine Summe beschreiben:

1. L : Mittlere Codewortlänge
2. s_i : Das Ausgangssymbol
3. l_i : Länge von s_i
4. p_i: Wahrscheinlichkeit für Ausgangssymbol oder Wort

$$L = \sum_{i=1}^{n} p_i * l_i.$$

Ein wichtiges Merkmal beim Huffman Code ist, dass häufiger vorkommende Wörter(p_i ist dabei größer) kürzere Codes haben. Durch den Huffman Algorithmus gilt dann für: $s_1, s_2, s_3, \ldots, s_n$:

- $p_1 \geq p_2 \geq p_3 \geq \ldots \geq p_n$
- $l_1 \leq l_2 \leq\leq l_3 \leq \ldots \leq l_n$

$\Rightarrow$ Die durch den Huffman Algorithmus vorgelegte aufsteigende Sortierung macht den Code optimal.

6 Verbesserungen

[2]

Trotzt dessen, dass der Huffman Code eine gute Kompression bietet, ist diese nicht optimal. Dies ist zum Beispiel der Fall, wenn die Anzahl der häufigsten Buchstaben P_{max} sehr hoch ist.

Zeichen	Code
a_1	0
a_2	11
a_3	10

6.1 Beispiel

Man nehme ein Text mit den Zeichen $A = a_1, a_2, a_3$ und die Wahrscheinlichkeiten: $P(a_1) = 0.8, P(a_2) = 0.02 \quad \& \quad P(a_3) = 0.18$. Die dazugehörige Huffman Codierung würde Folgendermaßen aussehen:

Die durchschnittliche Codelänge beträgt dabei:

$$L(C) = (\sum_{i=1}^{3})/3 = 1,2\frac{bits}{symbol}.$$

Die Entropie[4] oder der Informationsgehalt für den Text A beträgt dabei:

$$H(X) = -\sum_{i=1}^{3} p_i * \log_2(p_i) = 0,82\frac{bits}{symbol}.$$

Die Redundanz für diesen Code(Differenz von $L(C)$ und $H(X)$) wäre dann:

$$L(C) - H(X) = 1,2 - 0,82 = 0,384\frac{bits}{symbol}.$$

Wenn man das Ergebnis mit dem eigentlichen Informationsgehalt vergleicht, kann man erkennen dass der Code 47% mehr Bits verbraucht als eigentlich nötig.

6.1.1 Erweiterter Huffman Code

Um die Redundanz im Huffman Code zu reduzieren, kann man anstatt ein Codewort für jedes Zeichen, ein Codewort für jeweils zwei Zeichen erstellen:

Zeichen	Wahrscheinlichkeit	Code
a_1a_1	0,64	0
a_1a_2	0,016	10101
a_1a_3	0,144	11
a_2a_1	0,016	10100
a_2a_2	0,0004	10100101
a_2a_3	0,0036	10010011
a_3a_1	0,1440	100
a_3a_2	0,0036	10100100
a_3a_3	0,0324	10100

1. $L(C)_1 = 1,7228\frac{bits}{symbol}$
2. $L(C) = \frac{L(C)_1}{2} = \frac{1,7228}{2} = 0,8614$

Da ein Symbol im erweiterten Huffman Code zwei Zeichen entspricht, ist die durchschnittliche Wortlänge kürzer. Dies hat wiederum eine kleinere Redundanz zu Folge die nur bei $0,045\frac{bits}{Symbol}$ liegt. Das wären nur $5,5\%$ des Informationsgehaltes und deshalb viel Redundanzfreier als der klassische Huffman Code.

Literaturverzeichnis

[1] Die JPEG-Kompression - von S.Wickenburg, A.Rooch und J.GroSS.

[2] Extended huffman coding - http://www.ques10.com/p/30204/explain-extended-huffman-coding-what-is-the-advant/.

[3] Huffman-code - Hochschule flensburg.

[4] Entropie (Informationstheorie - Wikipedia), Feb. 2019. Page Version ID: 185850842.

[5] Huffman-Kodierung - Wikipedia, May 2019. Page Version ID: 188812775.