2. Woche
Eindeutige Entschlüsselbarleit, Sätze von Kraft und McMillan, Huffmancodierung

Zwei Beispiele

$$C: \begin{cases} a & \mapsto & 0 \\ b & \mapsto & 01 \\ c & \mapsto & 011 \\ d & \mapsto & 0111 \end{cases} \quad \text{und} \quad C': \begin{cases} a & \mapsto & 0 \\ b & \mapsto & 10 \\ c & \mapsto & 110 \\ d & \mapsto & 1110 \end{cases}$$

Beide Codes eindeutig entschlüsselbar.

Der erste ist nicht sofort entschlüsselbar.

Der zweite ist ein Präfixcode, d.h. sofort entschlüsselbar.

Aber, sie sind die Spiegelung von einenander. Vielleicht sollen wir nicht nur analysieren, wie Codeworte anfangen (Präfixe), aber auch wie sie enden ($\Rightarrow$ Suffixe), um die eindeutige Entschlüsselbarkeit zu charakterisieren.

Wann sind Codes eindeutig entschlüsselbar?

Definition Suffix

Sei C ein Code. Eine Folge $s \in \{0,1\}^*$ heißt Suffix in C falls eine der drei folgenden Bedingungen erfüllt ist:

1. $\exists c_i, c_j \in C : c_i = c_j s$ oder
2. $\exists c \in C$ und einen Suffix s' in C: $s' = cs$ oder
3. $\exists c \in C$ und einen Suffix s' in C: $c = s's$.

Bedeutung der Bedingungen

- Bedingung 1: Codewort c_j lässt sich zu Codewort c_i erweitern.
- Bedingung 2: Codewort c lässt sich zu Suffix s' erweitern.
- Bedingung 3: Suffix s' lässt sich zu Codewort c erweitern.

Effiziente Berechnung von Suffixen

Algorithmus Berechnung Suffix

EINGABE: $C = \{c_1, \ldots, c_n\}$

1. Setze $S := \emptyset$, $T := \emptyset$.
2. Für alle $c_i, c_j \in C \times C$: Falls es ein $s \in \{0,1\}^*$ gibt mit $c_i = c_j s$, füge s in S und T ein.
3. Solange $T \neq \emptyset$
   1. Entferne ein beliebiges s' aus T.
   2. Für alle $c \in C$: Falls es ein $s \in \{0,1\}^* \setminus S$ gibt mit

      $$s' = c' s \quad \text{oder}$$
      $$c = s' s \,,$$

      dann füge s zu S und T hinzu.

AUSGABE: Menge S der Suffixe von C

Laufzeit Suffixberechnung

Laufzeit:

- Schritt 2: $\mathcal{O}(n^2)$ Codewortpaare
- Suffixlänge ist durch $\max_i\{|c_i|\}$ beschränkt.
- Es kann höchstens $n \cdot \max_i\{|c_i|\}$ Suffixe geben.
- Schritt 3: $\mathcal{O}(n^2 \cdot \max_i\{|c_i|\})$
- **Polynomiell in der Eingabelänge**: n, $max_i\{|c_i|\}$

Beispiele Suffixberechnung

- Code $C_2 = \{0, 1, 00\}$
  - Suffix $s_1 = 0$, denn $c_3 = c_1 0$.
- Code $C_3 = \{0, 01, 011\}$
  - Suffix $s_1 = 1$, denn $c_2 = c_1 1$.
  - Suffix $s_2 = 11$, denn $c_3 = c_1 11$.
- Code $C_4 = \{0, 10, 11\}$
  - Keine Suffixe, da Präfixcode.
- Code $C_5 = \{1, 110, 101\}$
  - Suffix $s_1 = 10$, denn $c_2 = c_1 10$.
  - Suffix $s_2 = 01$, denn $c_3 = c_1 01$.
  - Suffix $s_3 = 0$, denn $s_1 = c_1 0$.
  - Suffix $s_4 = 1$, denn $c_3 = s_1 1$.

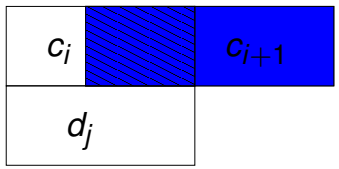
Kriterium für eindeutig entschlüsselbar

Satz Eindeutig entschlüsselbar

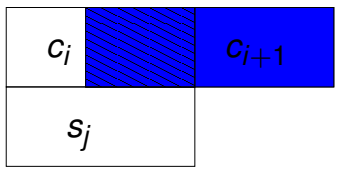
C ist ein eindeutig entschlüsselbarer Code $\Leftrightarrow$ Kein Suffix ist Codewort in C.

z.z.: C nicht eindeutig entschlüsselbar $\Rightarrow$ Suffix ist Codewort

- Zwei gleiche Folgen $c_1 \ldots c_n$ und $d_1 \ldots d_m$ von verschiedenen Codeworten
- Fall 1: Codewort c_i lässt sich zu d_j erweitern

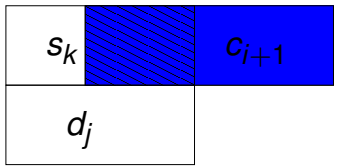


- Fall 2: Codewort c_i lässt sich zu Suffix s_j erweitern



Suffix ist Codewort

- Fall 3: Suffix s_k lässt sich zu Codewort d_j erweitern



- Nach jedem Schritt beginnt der konstruierte Suffix mit einem Codewortpräfix.
- Der zuletzt konstruierte Suffix ist identisch mit dem letzten Codewort von beiden Sequenzen.

Rückrichtung

z.z.: Suffix s ist ein Codewort $\Rightarrow$ C ist nicht eindeutig entschlüsselbar

- Suffix s ist aus Anwendungen der drei Regeln entstanden.
- Berechne die Kette zurück, aus der s entstanden ist.
  - Setze String $c^* \leftarrow s$. Iteriere:
  - 1. Fall $c_i = c_j s$: $c^* \leftarrow c_j c^*$, terminiere.
  - 2. Fall $s' = cs$: $c^* \leftarrow cc^*$, $s \leftarrow s'$.
  - 3. Fall $c = s's$: $c^* \leftarrow s'c^*$, $s \leftarrow s'$.
- Kette muss mit 1. Fall $c_i = c_j s'$ terminieren.
- Zwei verschiedene Entschlüsselungen:
  Eine beginnt mit c_i, die andere mit c_j.
- Beide sind gültig, da der letzte Suffix ein Codewort ist.

Beispiel: Für $C = 1, 110, 101$ erhalten wir für den Suffix 1 den String $c^* = 1101$ mit gültigen Decodierungen $1|101$ und $110|1$.

Sätze von Kraft und McMillan

Satz von Kraft (1949)

Ein Präfixcode C für das Alphabet $A = \{a_1, \ldots, a_n\}$ mit Codierungslängen $|C(a_j)| = \ell_j$ existiert gdw

$$\sum_{j=1}^{n} \frac{1}{2^{\ell_j}} \leq 1 \ .$$

Diese Ungleichung heißt *Krafts Ungleichung*.

Satz von McMillan (1956)

Falls der Code C für das Alphabet $A = \{a_1, \ldots, a_n\}$ mit Codierungslängen $|C(a_j)| = \ell_j$ eindeutig entschlüsselbar ist, dann

$$\sum_{j=1}^{n} \frac{1}{2^{\ell_j}} \leq 1 \ .$$

Präfixcodes genügen

Korollar

Ein Präfixcode C existiert gdw es einen eindeutig entschlüsselbaren Code C mit denselben Codierungslängen gibt.

- Wir zeigen den Ringschluss für:
  $\sum_{j=1}^{n} 2^{-\ell_j} \leq 1 \Rightarrow$ Präfix $\Rightarrow$ Eindeutig entschlüsselbar

  (Präfix $\Rightarrow$ Eindeutig entschlüsselbar: letzte Vorlesung)
- Gegeben sind Codierungslängen ℓ_j.
- Gesucht ist ein Präfixcode mit $\ell_j = |C(a_j)|$.
- Definiere $\ell := \max\{\ell_1, \ldots, \ell_n\}$, $n_i :=$ Anzahl ℓ_j mit $\ell_j = i$.

$$\sum_{j=1}^{n} 2^{-\ell_j} = \sum_{j=1}^{\ell} n_j 2^{-j} \leq 1.$$

Beweis: $\sum_{j=1}^{n} 2^{-\ell_j} \leq 1 \Rightarrow$ Präfix

Induktion über ℓ:

- **IV** $\ell = 1$**:** $n_1 \leq 2 \Rightarrow$ Präfixcode $C \subseteq \{0,1\}$, max 2 Codeworte, konstruierbar.
- **IA** Falls $\sum_{i=1}^{\ell-1} n_i/2^i \leq 1$ dann $\exists$ Präfixcode mit diesen Codewortlängen.
- **IS** $\ell - 1 \to \ell$**:** Sei $\sum_{i=1}^{\ell} n_i/2^i \leq 1$, d.h. $n_\ell \leq 2^\ell - n_1 2^{\ell-1} - n_2 2^{\ell-2} - \ldots n_{\ell-1} 2$.
  $2^\ell - n_1 2^{\ell-1} - n_2 2^{\ell-2} - \ldots n_{\ell-1} 2$ ist der max. Wert von n_ℓ.
- Aus **IA** $\exists$ Präfixcode C' mit n_i Worten der Länge i, $i = 1, \ldots, \ell - 1$.
- Anzahl der Worte der Länge ℓ: 2^ℓ
- Wir zählen die durch C' ausgeschlossenen Worte der Länge ℓ.
  Sei $c_i \in C'$ mit Länge ℓ_i. Dann enthalten alle $c_i s \in \{0,1\}^\ell$ mit beliebigem $s \in \{0,1\}^{\ell-\ell_i}$ den Präfix c_i.
  - Durch Präfixe der Länge 1 ausgeschlossene Worte: $n_1 \cdot 2^{\ell-1}$ — Menge M_1;
  - Durch Präfixe der Länge 2 ausgeschlossene Worte: $n_2 \cdot 2^{\ell-2}$ — Menge M_2;

  $\vdots$
  - Durch Präfixe der Länge $\ell - 1$ ausgeschlossene Worte: $n_{\ell-1} \cdot 2$ — Menge $M_{\ell-1}$.

  N.B. Die Mengen M_i sind paarweise disjunkt.

$\Rightarrow$ wir können bis $2^\ell - (n_1 2^{\ell-1} + \cdots + n_{\ell-1} 2)$ Symbole mit den verbleibenden Worten der Länge ℓ codieren, und die Präfixcode-Eigenschaft erhalten.

- Das dies der max. Wert von n_ℓ ist, dies ist immer möglich.

Eindeutig entschlüsselbar $\Rightarrow \sum_{j=1}^{n} 2^{-\ell_j} \leq 1$

- Sei C eindeutig entschlüsselbar mit $C(a_j) = \ell_j$, $\ell = max_j\{\ell_j\}$.
- Wählen $r \in \mathbb{N}$ beliebig. Betrachten

$$\left(\sum_{j=1}^{n} 2^{-\ell_j}\right)^r = \sum_{i=1}^{r\ell} m_i 2^{-i}$$

- Analog zum Beweis zuvor: m_i = Anzahl Strings aus $\{0,1\}^i$, die sich als Folge von r Codeworten schreiben lässt.
- C eindeutig entschlüsselbar: Jeder String aus $\{0,1\}^i$ lässt sich als höchstens eine Folge von Codeworten schreiben, d.h. $m_i \leq 2^i$.
- Damit gilt $\sum_{i=1}^{r\ell} m_i 2^{-i} \leq r\ell \quad \Rightarrow \quad \sum_{j=1}^{n} 2^{-\ell_j} \leq (r\ell)^{\frac{1}{r}}$
- Für $r \to \infty$ folgt $\sum_{j=1}^{n} 2^{-\ell_j} \leq 1$.

Effiziente (bandbreitensparende) Codierung

Szenario: Codierung mit variabler Codewortlänge.

Intuition: Je wahrscheinlicher ein Symbol ist, desto kürzer soll seine Codierung sein.

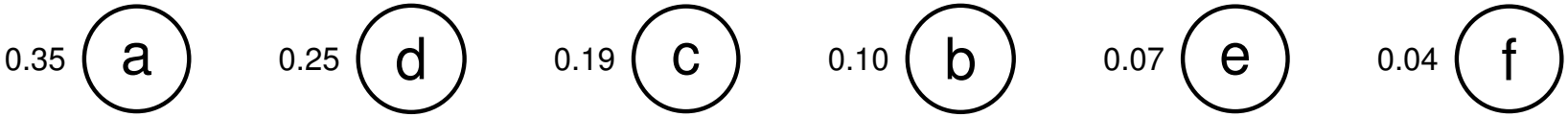
Huffman Codierung macht genau das.
Erst beschreiben wir sie, dann beweisen dass der resultierende Code kompakt ist.

Beispiel

Sei folgende Quelle gegeben mit Quellenwahrscheinlichkeiten

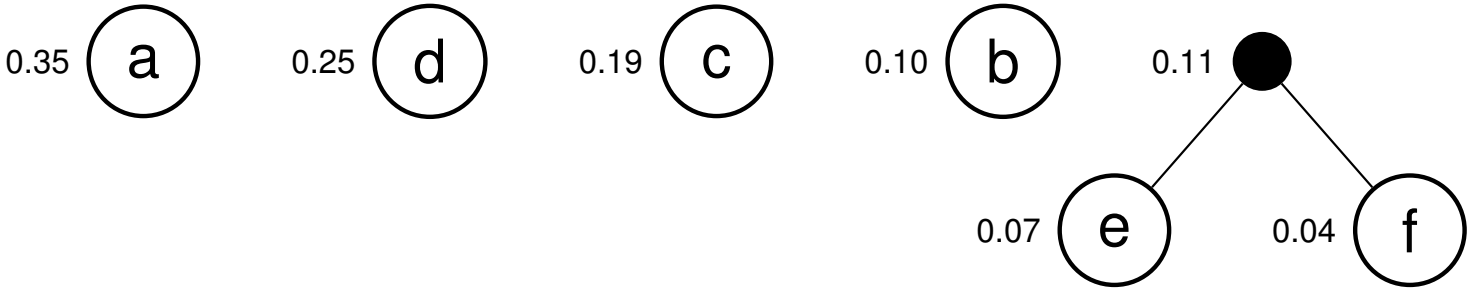
Symbol	*Wahrscheinlichkeit*
a	0.35
b	0.10
c	0.19
d	0.25
e	0.07
f	0.04

Beispiel



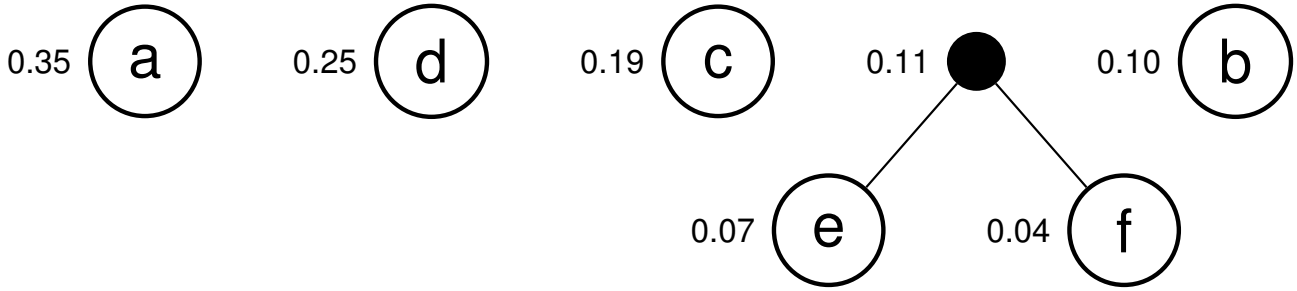
Schritt 1: Bilde sechs Knoten, etikettiert je mit einem Symbol. Die Knoten sind nach absteigender Ws des entsprechenden Symbols geordnet.
Diese sind die „Level 1“ Knoten.
Neben jedem Knoten schreiben wir die Ws.

Beispiel



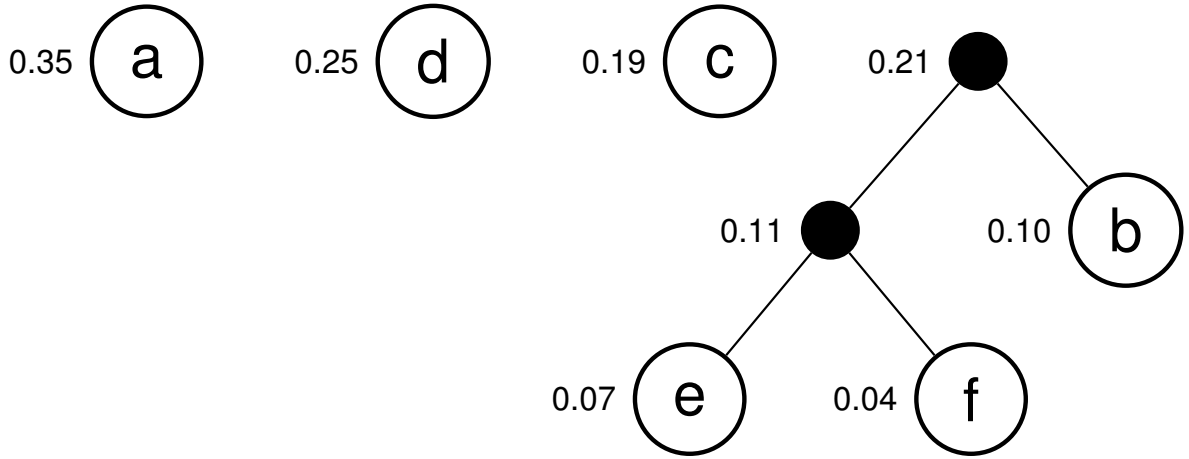
Schritt 2: Verbinde die zwei Knoten nach rechts (Symbole mit niedrigsten Ws) einem neuen Knoten dessen Ws ist die Summe der Ws der Zwei alten Knoten.
Die alten Knoten werden Kinder des neuen Knotens.
Der neue Knoten wird sich in die „Level 1“ Knoten einreihen.

Beispiel



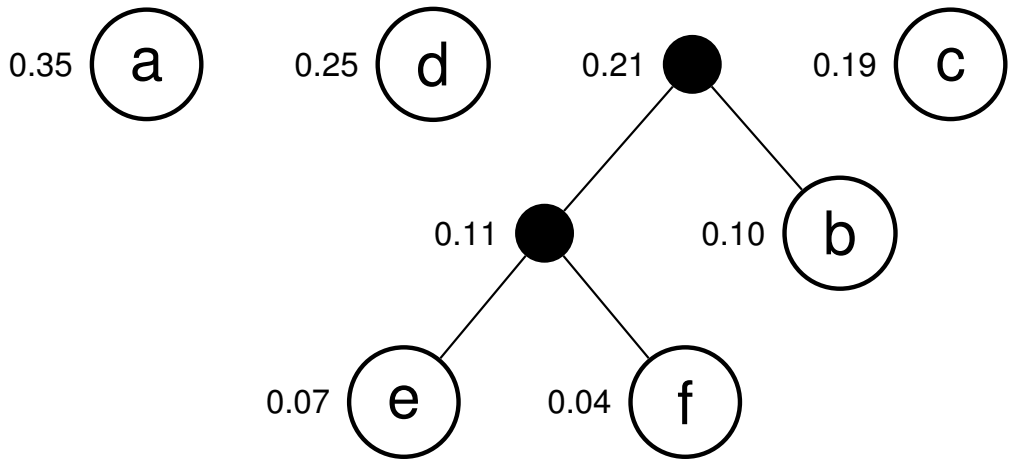
Schritt 3: Ordne die „Level 1“ Knoten nach absteigender Wahrscheinlichkeit wieder.

Beispiel



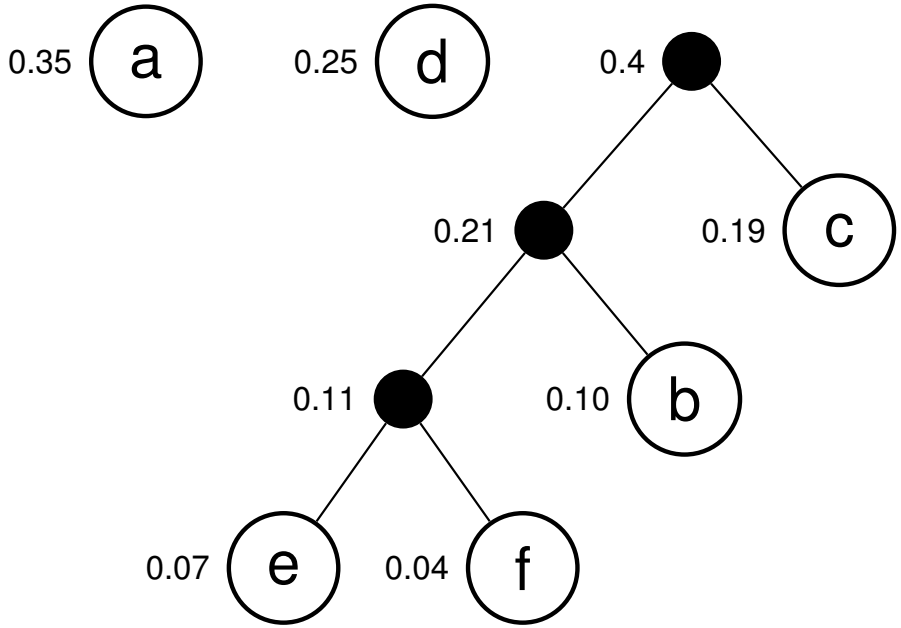
Verbinde die zwei Knoten nach rechts (also die mit den niedrigsten Wahrscheinlichkeiten) wie in Schritt 2.

Beispiel



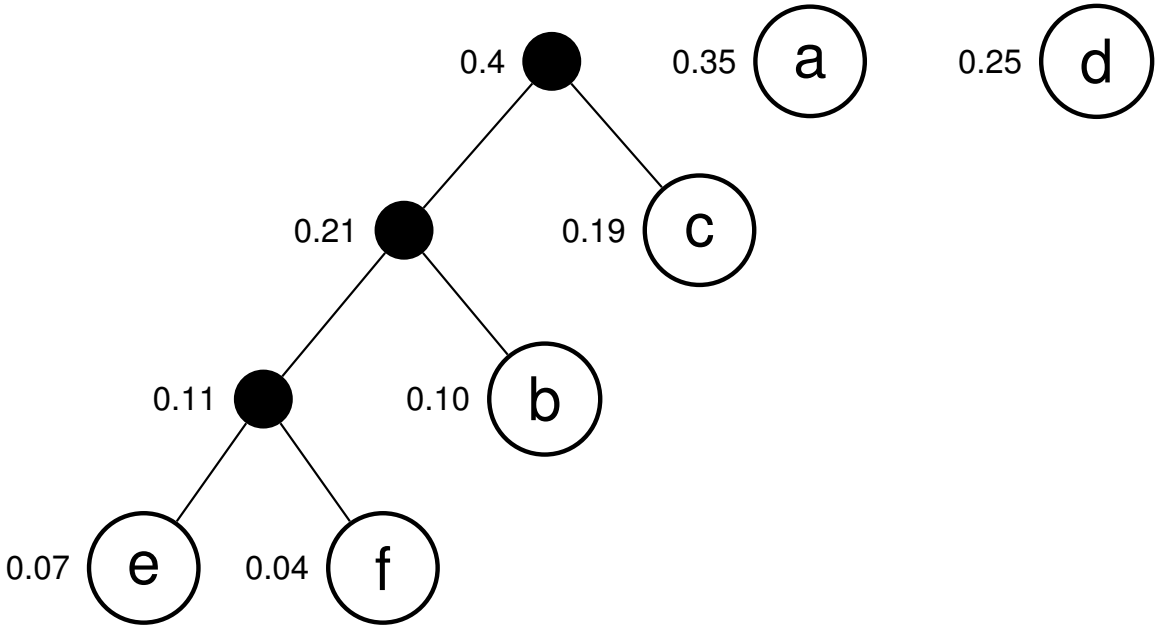
Und ordne wieder, wie in Schritt 3.

Beispiel



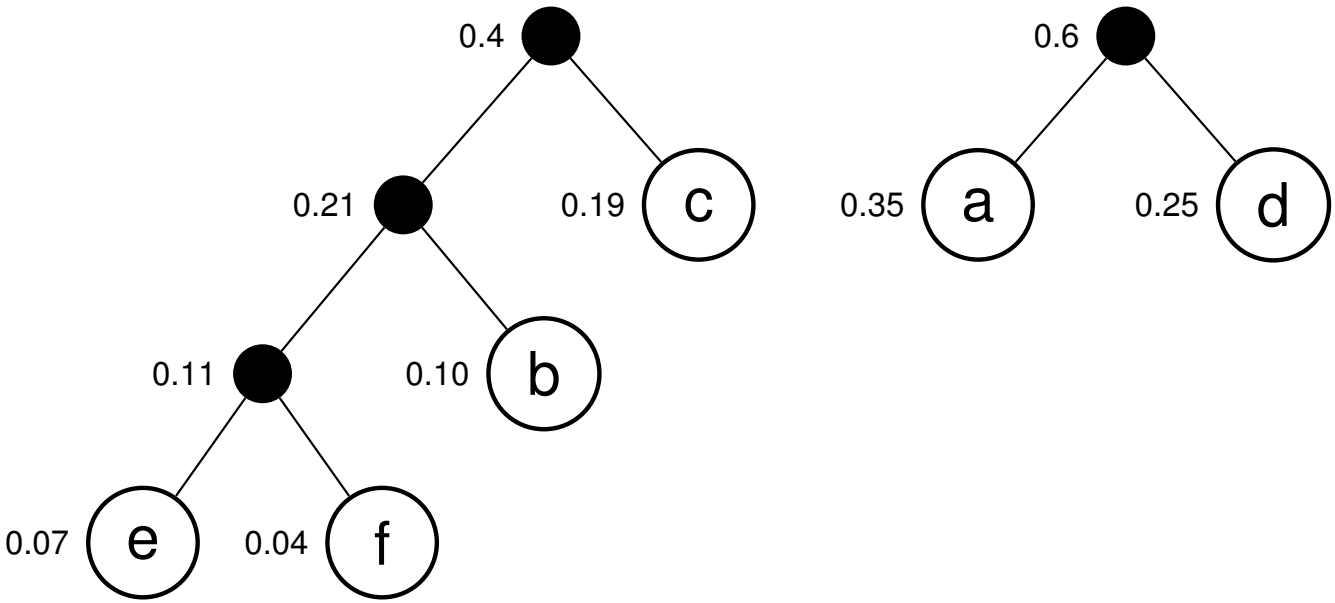
Verbinde.

Beispiel



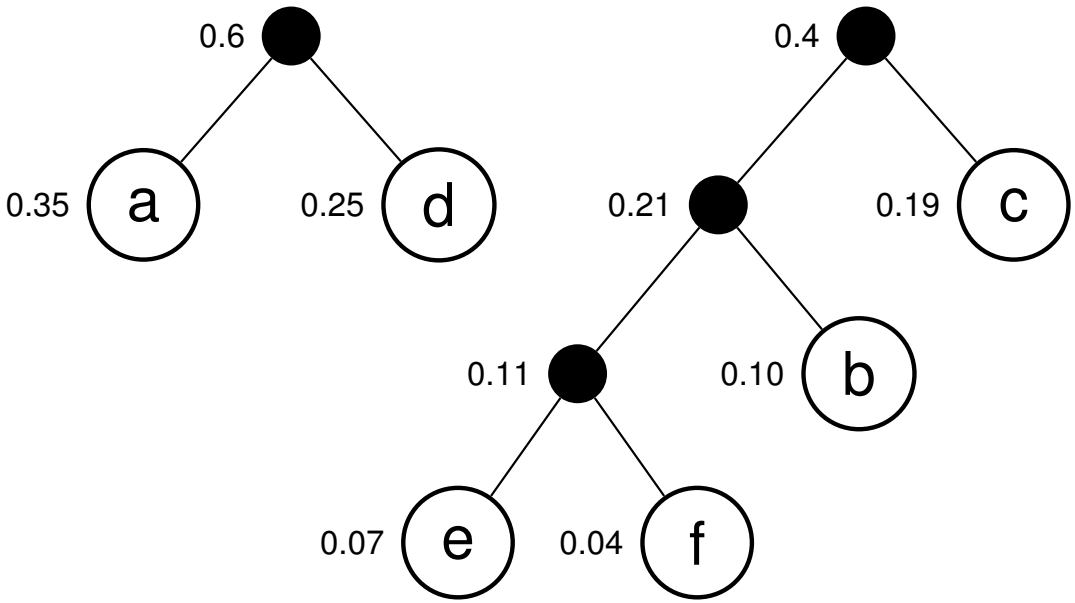
Ordne.

Beispiel



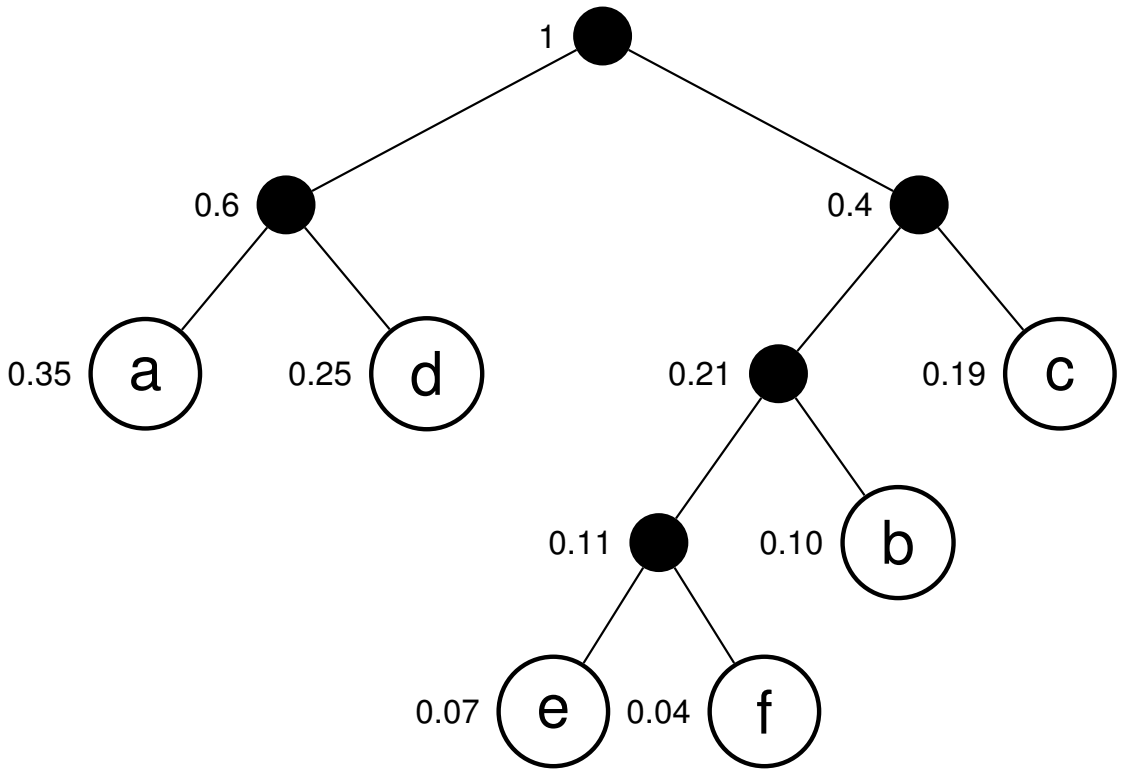
Verbinde.

Beispiel



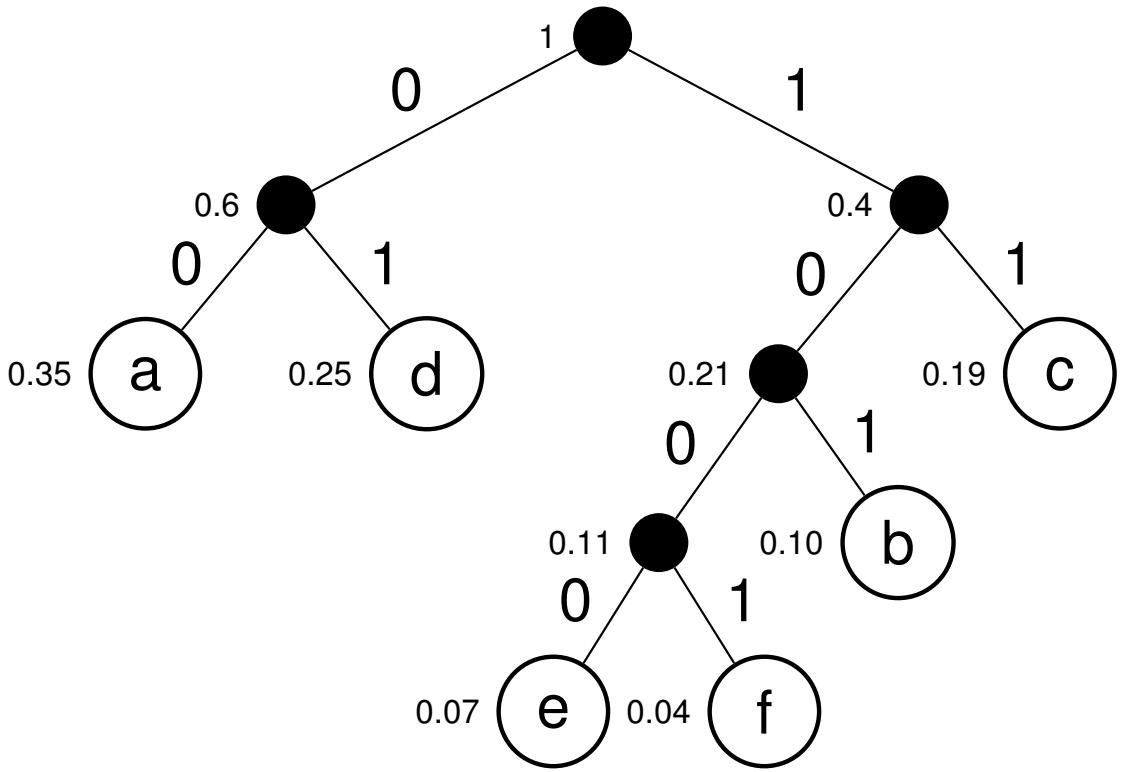
Ordne.

Beispiel



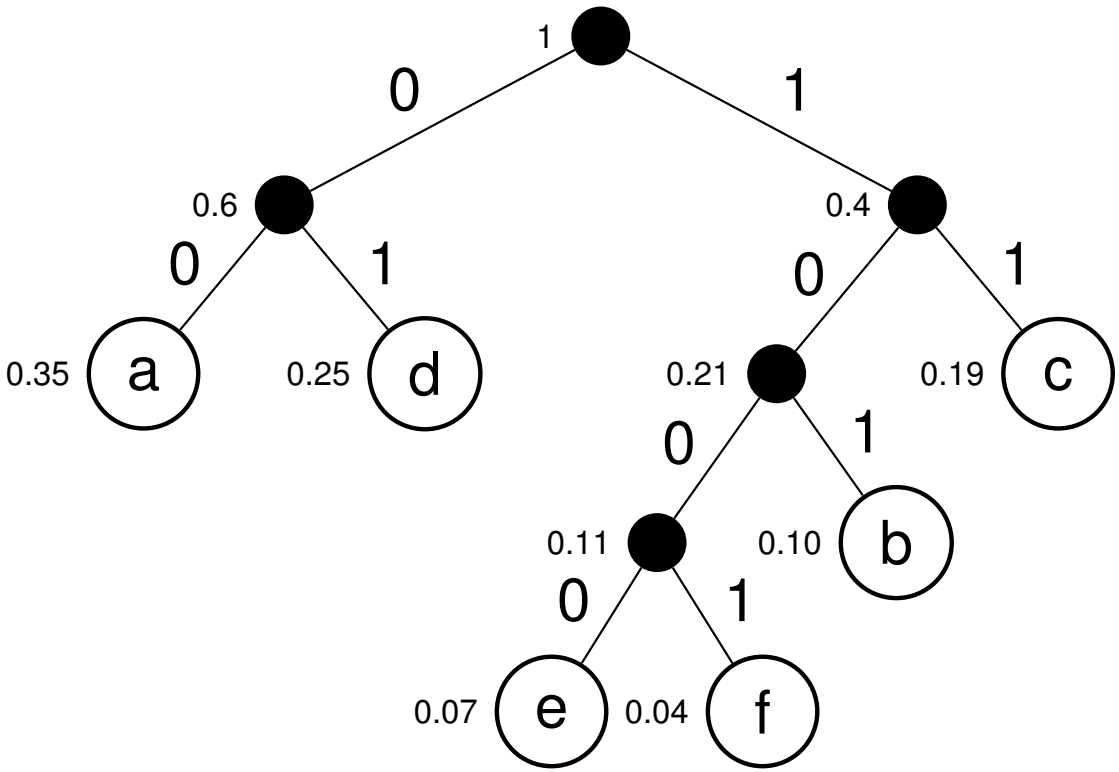
Verbinde. Nun haben wir einen Baum.

Beispiel



Etikettiere die Kanten. Kanten nach links mit 0, die nach rechts mit 1.

Beispiel



$$\begin{cases} a & \mapsto & 00 \\ b & \mapsto & 101 \\ c & \mapsto & 11 \\ d & \mapsto & 01 \\ e & \mapsto & 1000 \\ f & \mapsto & 1001 \end{cases}$$

Lese den Code ab! Das ist ein Präfixcode, denn die Labels mit den Quellensymbolen befinden sich nur an Blättern.

Beobachtung

Symbol	*Wahrscheinlichkeit*
a	0.35
b	0.10
c	0.19
d	0.25
e	0.07
f	0.04

$$\Rightarrow \quad \begin{cases} a & \mapsto & 00 \\ b & \mapsto & 101 \\ c & \mapsto & 11 \\ d & \mapsto & 01 \\ e & \mapsto & 1000 \\ f & \mapsto & 1001 \end{cases}$$

Man sieht, dass die Symbole, die am häufigsten vorkommen, nicht längere Codierung besitzen als Symbole, die seltener vorkommen.

Mit anderen Worten: **Falls $p_i > p_j$, dann ist $\ell_i \leq \ell_j$.**

Huffman Codierung

Szenario: Quelle Q mit Symbole $\{a_1, \ldots, a_n\}$

- a_i sortiert nach absteigenden Quellws. $p_1 \geq p_2 \geq \cdots \geq p_n$.

Algorithmus Huffman-Codierung

Eingabe: Symbole a_i mit absteigend sortierten p_i, $i = 1, \ldots, n$.

1. IF (n=2), Ausgabe $C(a_1) = 0$, $C(a_2) = 1$.
2. ELSE
   1. Bestimme $k \in \mathbb{Z}_{n-1}$ mit $p_k \geq p_{n-1} + p_n \geq p_{k+1}$.
   2. $(p_1, \ldots, p_k, p_{k+1}, p_{k+2}, \ldots, p_{n-1}) \leftarrow$
      $(p_1, \ldots, p_k, p_{n-1} + p_n, p_{k+1}, \ldots, p_{n-2})$
   3. $(C(a_1), \ldots, C(a_{k-1}), C(a_{k+1}), \ldots, C(a_{n-2}), C(a_k)0, C(a_k)1) \leftarrow$
      Huffmann-Codierung$(a_1, \ldots, a_{n-1}, p_1, \ldots, p_{n-1})$

Ausgabe: kompakter Präfixcode für Q

Laufzeit: $\mathcal{O}(n^2)$ ($\mathcal{O}(n \log n)$ mit Hilfe von Heap-Datenstruktur)

Eigenschaften kompakter Codes

Sei $\ell_i := |C(a_i)|$.

Lemma Eigenschaften kompakter Codes

Sei C ein kompakter Code, oBdA ist C ein Präfixcode.

1. Falls $p_i > p_j$, dann ist $\ell_i \leq \ell_j$
2. Es gibt mindestens zwei Codeworte in C mit maximaler Länge.
3. Unter den Worten mit maximaler Länge existieren zwei Worte, die sich nur in der letzten Stelle unterscheiden.

Beweis der Eigenschaften

Beweis:

1. Sei $\ell_i > \ell_j$. Dann gilt

$$\begin{aligned} & p_i\ell_i + p_j\ell_j = p_i(\ell_i - \ell_j + \ell_j) + p_j(\ell_j - \ell_i + \ell_i) \\ = \quad & p_i\ell_j + p_j\ell_i + (\ell_i - \ell_j)(p_i - p_j) > p_i\ell_j + p_j\ell_i \end{aligned}$$

   D.h. vertauschen der Codierungen von a_i und a_j verkürzt den Code.

2. Sei $c = c_1 \dots c_n \in C$ das einzige Codewort mit maximaler Länge. Streichen von c_n führt zu einem Präfixcode mit kürzerer erwarteter Codewortlänge.
3. Annahme: Alle Paar von Codeworten maximaler Länge unterscheiden sich nicht nur in der letzten Komponente.
   - Entferne die letzte Komponente eines beliebigen Codewortes maximaler Länge.
   - Wir erhalten einen Präfixcode mit kürzerer Länge.

Optimalität der Huffman-Codierung

Satz

Die Huffman-Codierung liefert einen kompakten Code.

Beweis per Induktion über n.

- **IA:** $n = 2$: Für $\{a_1, a_2\}$ ist die Codierung $\{0, 1\}$ kompakt.
- **IS:** $n - 1 \to n$: Sei C' kompakt für $\{a_1, \ldots, a_n\}$.
  - Lemma,2: C' enthält zwei Codeworte maximaler Länge.
  - Lemma,3: Unter den Codeworten maximaler Länge gibt es zwei Codeworte $c0, c1 \in C'$ mit $c \in \{0, 1\}^*$, die sich nur in der letzten Stelle unterscheiden.
  - Lemma,1: Die beiden Symbole a_{n-1}, a_n mit kleinster Quellws besitzen maximale Codewortlänge. Vertausche die Codierungen dieser Symbole mit $c0, c1$.
  - a_{n-1} oder a_n tauchen mit Ws $p_{n-1} + p_n$ auf.
  - **IA:** Huffman-Codierung liefert kompakten Präfixcode C für $a_1, \ldots, a_{n-2}, a'$ mit Quellws $p_1, \ldots, p_{n-2}, p_{n-1} + p_n$
  - $C(a_1), \ldots, C(a_{n-2}), C(a')0 = c0, C(a')1 = c1$ ist Präfixcode mit erwarteter Codewortlänge $E(C')$, d.h. die Huffman-Codierung liefert einen kompakten Präfixcode.