

*„Ich hoffe, es werden Ew. Hochfürstliche Durchlaucht in Gnaden vermerken, daß ich sowohl dem Gebrauche, als meinem Gemüths=Triebe zu Folge, bei dem eingetretenen neuen Jahre, auf dieses und viele folgende, Denenselben in beständiger Gesundheit alle selbst verlangende hohe Fürstliche Ersprießlichkeit zu gemeinem und Dero Lande besondern Besten, aus treuem Herzen anwünsche.“*

schrrieb in einem Brief datiert auf den 2. Januar 1697 an den Herzog von Braunschweig-Wolfenbüttel

*„Ich hoffe, es werden Ew. Hochfürstliche Durchlaucht in Gnaden vermerken, daß ich sowohl dem Gebrauche, als meinem Gemüths=Triebe zu Folge, bei dem eingetretenen neuen Jahre, auf dieses und viele folgende, Denenselben in beständiger Gesundheit alle selbst verlangende hohe Fürstliche Ersprießlichkeit zu gemeinem und Dero Lande besondern Besten, aus treuem Herzen anwünsche.“*

schrrieb in einem Brief datiert auf den 2. Januar 1697 an den Herzog von Braunschweig-Wolfenbüttel



der Philosoph, Mathematiker, Physiker,  
Bibliothekar, ...  
Gottfried Wilhelm Leibniz (1646–1716)

# Grundbegriffe der Informatik

## Einheit 10: Codierungen

Thomas Worsch

Karlsruher Institut für Technologie, Fakultät für Informatik

Wintersemester 2010/2011

## Codierungen

### Von Wörtern zu Zahlen und zurück

- Dezimaldarstellung von Zahlen

- Andere Zahldarstellungen

- Von Zahlen zu ihren Darstellungen

### Von einem Alphabet zum anderen

- Ein Beispiel: Übersetzung von Zahldarstellungen

- Homomorphismen

- Beispiel Unicode: UTF-8

### Huffman-Codierung

- Algorithmus zur Berechnung von Huffman-Codes

- Weiteres zu Huffman-Codes

## Codierungen

### Von Wörtern zu Zahlen und zurück

- Dezimaldarstellung von Zahlen

- Andere Zahldarstellungen

- Von Zahlen zu ihren Darstellungen

### Von einem Alphabet zum anderen

- Ein Beispiel: Übersetzung von Zahldarstellungen

- Homomorphismen

- Beispiel Unicode: UTF-8

### Huffman-Codierung

- Algorithmus zur Berechnung von Huffman-Codes

- Weiteres zu Huffman-Codes

- ▶ aus Indien
- ▶ Ziffern des Alphabetes  $Z_{10} = \{0, \dots, 9\}$ .
- ▶ Bedeutung  $\text{num}_{10}(x)$  einer einzelnen Ziffer  $x$  als Zahl:

$x$	0	1	2	...	8	9
$\text{num}_{10}(x)$	<i>null</i>	<i>eins</i>	<i>zwei</i>	...	<i>acht</i>	<i>neun</i>

- ▶ aus Indien
- ▶ Ziffern des Alphabetes  $Z_{10} = \{0, \dots, 9\}$ .
- ▶ Bedeutung  $\text{num}_{10}(x)$  einer einzelnen Ziffer  $x$  als Zahl:

---

$x$	0	1	2	3	4	5	6	7	8	9
$\text{num}_{10}(x)$	0	1	2	3	4	5	6	7	8	9

---

- ▶ Bedeutung eines Wortes  $x_{k-1} \cdots x_0 \in Z_{10}^*$  von Ziffern

$$\text{Num}_{10} : Z_{10}^* \rightarrow \mathbb{N}_0$$

- ▶ aus Indien
- ▶ Ziffern des Alphabetes  $Z_{10} = \{0, \dots, 9\}$ .
- ▶ Bedeutung  $\text{num}_{10}(x)$  einer einzelnen Ziffer  $x$  als Zahl:

$x$	0	1	2	3	4	5	6	7	8	9
$\text{num}_{10}(x)$	0	1	2	3	4	5	6	7	8	9

- ▶ Bedeutung eines Wortes  $x_{k-1} \cdots x_0 \in Z_{10}^*$  von Ziffern

$$\text{Num}_{10} : Z_{10}^* \rightarrow \mathbb{N}_0$$

- ▶ Schule:

$$10^{k-1} \cdot \text{num}_{10}(x_{k-1}) + \cdots + 10^1 \cdot \text{num}_{10}(x_1) + 10^0 \cdot \text{num}_{10}(x_0)$$

- ▶ Pünktchenvermeidung:

$$\text{Num}_{10}(\varepsilon) = 0$$

$$\forall w \in Z_{10}^* \quad \forall x \in Z_{10} : \text{Num}_{10}(wx) = 10 \cdot \text{Num}_{10}(w) + \text{num}_{10}(x)$$

- ▶ geboren 1. Juli 1646 in Leipzig  
gestorben am 14. November 1716 in Hannover
- ▶ baute erste Maschine, die zwei Zahlen multiplizieren konnte
- ▶ in einem Brief vom 2. Januar 1697 an den Herzog von Braunschweig-Wolfenbüttel:
  - ▶ auch wenn man nur zwei Ziffern 0 und 1 hat, kann man nichtnegative Zahlen darstellen und vernünftig rechnen

Pour l'Addition  
par exemple. ☺

$\begin{array}{r l} 110 & 6 \\ 111 & 7 \\ \hline 1101 & 13 \end{array}$	$\begin{array}{r l} 101 & 5 \\ 1011 & 11 \\ \hline 10000 & 16 \end{array}$	$\begin{array}{r l} 1110 & 14 \\ 10001 & 17 \\ \hline 11111 & 31 \end{array}$
---	--	---

Pour la Sou-  
straction.

$\begin{array}{r l} 1101 & 13 \\ 111 & 7 \\ \hline 110 & 6 \end{array}$	$\begin{array}{r l} 10000 & 16 \\ 1011 & 11 \\ \hline 101 & 5 \end{array}$	$\begin{array}{r l} 11111 & 31 \\ 10001 & 17 \\ \hline 1110 & 14 \end{array}$
---	--	---

Bildquelle [http://commons.wikimedia.org/wiki/Image:Leibniz\\_binary\\_system\\_1703.png](http://commons.wikimedia.org/wiki/Image:Leibniz_binary_system_1703.png)

- ▶ Ziffernmenge  $Z_2 = \{0, 1\}$
- ▶ definiere:  $\text{num}_2(0) = 0$  und  $\text{num}_2(1) = 1$  und

$$\text{Num}_2(\varepsilon) = 0$$

$$\forall w \in Z_2^* \quad \forall x \in Z_2 : \text{Num}_2(wx) = 2 \cdot \text{Num}_2(w) + \text{num}_2(x)$$

- ▶ Z. B.:

$$\begin{aligned}\text{Num}_2(\mathbf{1101}) &= 2 \cdot \text{Num}_2(\mathbf{110}) + 1 \\ &= 2 \cdot (2 \cdot \text{Num}_2(\mathbf{11}) + 0) + 1 \\ &= 2 \cdot (2 \cdot (2 \cdot \text{Num}_2(\mathbf{1}) + 1) + 0) + 1 \\ &= 2 \cdot (2 \cdot (2 \cdot (2 \cdot \text{Num}_2(\varepsilon) + 1) + 1) + 0) + 1 \\ &= 2 \cdot (2 \cdot (2 \cdot (2 \cdot 0 + 1) + 1) + 0) + 1 \\ &= 2^3 \cdot 1 + 2^2 \cdot 1 + 2^1 \cdot 0 + 2^0 \cdot 1 \\ &= 13\end{aligned}$$

- ▶ Ziffern  $Z_{16} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$ .
- ▶ Ziffernwerte:

$x$	0	1	2	3	4	5	6	7
$\text{num}_{16}(x)$	0	1	2	3	4	5	6	7
$x$	8	9	A	B	C	D	E	F
$\text{num}_{16}(x)$	8	9	10	11	12	13	14	15

- ▶ Zuordnung von Wörtern zu Zahlen gegeben durch

$$\text{Num}_{16}(\varepsilon) = 0$$

$$\forall w \in Z_{16}^* \quad \forall x \in Z_{16} : \text{Num}_{16}(wx) = 16 \cdot \text{Num}_{16}(w) + \text{num}_{16}(x)$$

- ▶ die Alphabete  $Z_2$ ,  $Z_3$ , usw. sind nicht disjunkt
- ▶ Darstellungen mehrdeutig; Beispiel 111:
  - $\text{Num}_2(111)$  die Zahl sieben,
  - $\text{Num}_8(111)$  die Zahl dreiundsiebzig,
  - $\text{Num}_{10}(111)$  die Zahl einhundertelf und
  - $\text{Num}_{16}(111)$  die Zahl zweihundertdreiundsiebzig.
- ▶ in manchen Programmiersprachen
  - ▶ Präfix `0b` für Darstellungen zur Basis 2
  - ▶ Präfix `0o` für Darstellungen zur Basis 8
  - ▶ Präfix `0x` für Darstellungen zur Basis 16

- ▶ Man kann zu  $w \in Z_k^*$  die repräsentierte Zahl  $\text{Num}_k(w)$  berechnen.
- ▶ Umgekehrt geht es auch: zu  $n \in \mathbb{N}_0$  die sogenannte  **$k$ -äre Darstellung** berechnen
- ▶ Alphabet  $Z_k$  mit  $k$  Ziffern
  - ▶ Bedeutung: die Zahlen in  $\mathbb{G}_k$
  - ▶ für  $i \in \mathbb{G}_k$  sei  $\text{repr}_k(i)$  das entsprechende Zeichen
  - ▶  $\text{repr}_k$  ist also gerade die Umkehrfunktion zu  $\text{num}_k$
- ▶ Gesucht: eine Repräsentation von  $n \in \mathbb{N}_0$  als Wort  $w \in Z_k^*$  mit der Eigenschaft  $\text{Num}_k(w) = n$ 
  - ▶ für die naheliegende Definition von  $\text{Num}_k$
- ▶ gibt es immer, und zwar gleich unendlich viele passende  $w$ : wenn  $\text{Num}_k(w) = n$ , dann auch  $\text{Num}_k(0w) = n$

// Eingabe:  $n \in \mathbb{N}_0$

$y \leftarrow n$

$w \leftarrow \varepsilon$

$m \leftarrow \begin{cases} 1 + \lfloor \log_k(n) \rfloor & \text{falls } n > 0 \\ 1 & \text{falls } n = 0 \end{cases}$

**for**  $i \leftarrow 0$  **to**  $m - 1$  **do**

$r \leftarrow y \bmod k$

$w \leftarrow \text{repr}_k(r) \cdot w$      // Konkatination

$y \leftarrow y \text{div } k$

**od**

// am Ende:  $n = \text{Num}_k(w)$

- ▶ Nehme  $k = 10$  und  $n = 4711$
- ▶ Dann ist  $m = 4$
- ▶ für jedes  $i \in \mathbb{G}_5$  haben die Variablen  $r$ ,  $w$  und  $y$  nach  $i$  Schleifendurchläufen die folgenden Werte

$i$	4	3	2	1	0
$r$	4	7	1	1	
$w$	4711	711	11	1	$\varepsilon$
$y$	0	4	47	471	4711

- ▶ Schleifeninvariante  $y \cdot 10^i + \text{Num}_k(w) = n$  drängt sich auf
- ▶ Wenn man weiß, dass am Ende  $y = 0$  ist, ist man fertig.

- ▶ Algorithmus liefert kürzestes Wort  $w \in Z_k^*$  mit  $\text{Num}_k(w) = n$ .
- ▶ schreiben dafür  $\text{Repr}_k(n)$ .
- ▶ Es ist stets

$$\text{Num}_k(\text{Repr}_k(n)) = n .$$

- ▶ Beachte:
  - ▶ umgekehrt im allgemeinen

$$\text{Repr}_k(\text{Num}_k(w)) \neq w$$

- ▶ weil „überflüssige“ führende Nullen wegfallen.

## noch ein Beispiel (1)

- ▶ Ziffernmenge  $Z = \{-1, 0, 1\}$
- ▶ definiere

$$\text{num}(-1) = -1$$

$$\text{num}(0) = 0$$

$$\text{num}(1) = 1$$

- ▶ definiere  $\text{Num} : Z^* \rightarrow \mathbb{Z}$

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \quad \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ Z. B.:  $\text{Num}(-101) = -3^2 + 0 + 3^0 = -8$  und  
 $\text{Num}(1-101) = +3^3 - 3^2 + 0 + 3^0 = 19$

- ▶  $\text{num}(\bar{1}) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \quad \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
    - ▶  $\text{inv}(1) = \bar{1}$ ,  $\text{inv}(\bar{1}) = 1$  und  $\text{inv}(0) = 0$ .
    - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1\bar{1}01) = \bar{1}10\bar{1}$
- ▶ dann gilt für alle  $w \in Z^*$ :  $\text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .

## noch ein Beispiel (2)

- ▶  $\text{num}(\uparrow) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
  - ▶  $\text{inv}(1) = \uparrow$ ,  $\text{inv}(\uparrow) = 1$  und  $\text{inv}(0) = 0$ .
  - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1\uparrow 01) = \uparrow 10\uparrow$
- ▶ dann gilt für alle  $w \in Z^*$ :  $\text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .
- ▶ „Addition“ wie „in der Schule“:

$$\begin{array}{cccc} 1 & \uparrow & 0 & 1 \\ & \uparrow & 0 & 1 \end{array}$$

---

## noch ein Beispiel (2)

- ▶  $\text{num}(\uparrow) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
  - ▶  $\text{inv}(1) = \uparrow$ ,  $\text{inv}(\uparrow) = 1$  und  $\text{inv}(0) = 0$ .
  - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1\uparrow 01) = \uparrow 10\uparrow$
- ▶ dann gilt für alle  $w \in Z^*$ :  $\text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .
- ▶ „Addition“ wie „in der Schule“:

$$\begin{array}{rcccc} 1 & \uparrow & 0 & 1 & \\ & & \uparrow & 0 & 1 \\ & & & 1 & \\ \hline & & & & \uparrow \end{array}$$

## noch ein Beispiel (2)

- ▶  $\text{num}(\bar{1}) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \quad \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
  - ▶  $\text{inv}(1) = \bar{1}$ ,  $\text{inv}(\bar{1}) = 1$  und  $\text{inv}(0) = 0$ .
  - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1\bar{1}01) = \bar{1}10\bar{1}$
- ▶ dann gilt für alle  $w \in Z^*$ :  $\text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .
- ▶ „Addition“ wie „in der Schule“:

$$\begin{array}{rcccc} 1 & \bar{1} & 0 & 1 \\ & \bar{1} & 0 & 1 \\ & 0 & 1 & \\ \hline & & 1 & \bar{1} \end{array}$$

## noch ein Beispiel (2)

- ▶  $\text{num}(1) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
  - ▶  $\text{inv}(1) = 1$ ,  $\text{inv}(1) = 1$  und  $\text{inv}(0) = 0$ .
  - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1101) = 1101$
- ▶ dann gilt für alle  $w \in Z^* : \text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .
- ▶ „Addition“ wie „in der Schule“:

$$\begin{array}{rcccc} 1 & 1 & 0 & 1 & \\ & & 1 & 0 & 1 & \\ 1 & 0 & 1 & & & \\ \hline 1 & 1 & 1 & & & \end{array}$$

## noch ein Beispiel (2)

- ▶  $\text{num}(\uparrow) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
  - ▶  $\text{inv}(1) = \uparrow$ ,  $\text{inv}(\uparrow) = 1$  und  $\text{inv}(0) = 0$ .
  - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1\uparrow 01) = \uparrow 10\uparrow$
- ▶ dann gilt für alle  $w \in Z^*$ :  $\text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .
- ▶ „Addition“ wie „in der Schule“:

$$\begin{array}{rcccc} 1 & \uparrow & 0 & 1 \\ & \uparrow & 0 & 1 \\ \uparrow & 0 & 1 & \\ \hline 0 & 1 & 1 & \uparrow \end{array}$$

## noch ein Beispiel (2)

- ▶  $\text{num}(1) = -1$ ,  $\text{num}(0) = 0$  und  $\text{num}(1) = 1$  und

$$\text{Num}(\varepsilon) = 0$$

$$\forall w \in Z^* \quad \forall x \in Z : \text{Num}(wx) = 3 \cdot \text{Num}(w) + \text{num}(x)$$

- ▶ definiere  $\text{inv} : Z^* \rightarrow Z^*$ 
  - ▶  $\text{inv}(1) = 1$ ,  $\text{inv}(1) = 1$  und  $\text{inv}(0) = 0$ .
  - ▶  $\text{inv}(\varepsilon) = \varepsilon$  und  $\forall w \in Z^* : \forall x \in Z : \text{inv}(wx) = \text{inv}(w)\text{inv}(x)$
- also z. B.  $\text{inv}(1101) = 1101$
- ▶ dann gilt für alle  $w \in Z^*$ :  $\text{Num}(\text{inv}(w)) = -\text{Num}(w)$ .
- ▶ „Addition“ wie „in der Schule“:

$$\begin{array}{rccccr} 1 & 1 & 0 & 1 & 19 \\ & & 1 & 0 & 1 & -8 \\ \hline & & 1 & 0 & 1 & \\ \hline 0 & 1 & 1 & 1 & & 11 \end{array}$$

## Das sollten Sie mitnehmen:

- ▶ Umwandlungen zwischen Zahlen und Wörtern
- ▶ mal wieder Schleifen
  - ▶ in passenden Beispielen ist Schleifeninvariante gut zu sehen
- ▶ schon Leibniz kannte die Binärdarstellung

## Das sollten Sie üben:

- ▶ selber Zahlen verschieden repräsentieren
- ▶ Algorithmen auch in Randfällen ausprobieren

## Codierungen

Von Wörtern zu Zahlen und zurück

Dezimaldarstellung von Zahlen

Andere Zahldarstellungen

Von Zahlen zu ihren Darstellungen

Von einem Alphabet zum anderen

Ein Beispiel: Übersetzung von Zahldarstellungen

Homomorphismen

Beispiel Unicode: UTF-8

Huffman-Codierung

Algorithmus zur Berechnung von Huffman-Codes

Weiteres zu Huffman-Codes

- ▶ Betrachte die Funktion  $\text{Trans}_{2,16} = \text{Repr}_2 \circ \text{Num}_{16}$
- ▶  $\text{Trans}_{2,16} : Z_{16}^* \rightarrow Z_2^*$
- ▶ Z. B.

$$\begin{aligned}\text{Trans}_{2,16}(\mathbf{A3}) &= \text{Repr}_2(\text{Num}_{16}(\mathbf{A3})) \\ &= \text{Repr}_2(163) = \mathbf{10100011}\end{aligned}$$

- ▶ wesentlicher Punkt:
  - ▶  $\mathbf{A3}$  und  $\mathbf{10100011}$  haben *die gleiche Bedeutung*
  - ▶ nämlich die Zahl einhundertdreiundsechzig
- ▶ So etwas wollen wir eine **Übersetzung** nennen.

- ▶ Allgemein: Man schreibt Wörter aus einer formalen Sprache  $L_A \subseteq A^*$  und meint aber etwas anderes, ihre Bedeutung.
- ▶ Menge der Bedeutungen je nach Anwendungsfall sehr unterschiedlich
  - ▶ einfach: Zahlen,
  - ▶ kompliziert: Ausführung von Java-Programmen
  - ▶ im Folgenden schreiben wir einfach Sem für die Menge der „Bedeutungen“
- ▶ Gegeben:
  - ▶ zwei Alphabete  $A$  und  $B$
  - ▶  $L_A \subseteq A^*$  und  $L_B \subseteq B^*$
  - ▶ zwei Abbildungen  $\text{sem}_A : L_A \rightarrow \text{Sem}$  und  $\text{sem}_B : L_B \rightarrow \text{Sem}$
- ▶ Eine Abbildung  $f : L_A \rightarrow L_B$  heie eine **Übersetzung** bezüglich  $\text{sem}_A$  und  $\text{sem}_B$ , wenn  $f$  die Bedeutung erhält, d. h.

$$\forall w \in L_A : \text{sem}_A(w) = \text{sem}_B(f(w))$$

- ▶ Allgemein: Man schreibt Wörter aus einer formalen Sprache  $L_A \subseteq A^*$  und meint aber etwas anderes, ihre Bedeutung.
- ▶ Menge der Bedeutungen je nach Anwendungsfall sehr unterschiedlich
  - ▶ einfach: Zahlen,
  - ▶ kompliziert: Ausführung von Java-Programmen
  - ▶ im Folgenden schreiben wir einfach Sem für die Menge der „Bedeutungen“
- ▶ Gegeben:
  - ▶ zwei Alphabete  $A$  und  $B$
  - ▶  $L_A \subseteq A^*$  und  $L_B \subseteq B^*$
  - ▶ zwei Abbildungen  $\text{sem}_A : L_A \rightarrow \text{Sem}$  und  $\text{sem}_B : L_B \rightarrow \text{Sem}$
- ▶ Eine Abbildung  $f : L_A \rightarrow L_B$  heie eine **Übersetzung** bezüglich  $\text{sem}_A$  und  $\text{sem}_B$ , wenn  $f$  die Bedeutung erhält, d. h.

$$\forall w \in L_A : \text{sem}_A(w) = \text{sem}_B(f(w))$$

- ▶  $\text{Trans}_{2,16} = \text{Repr}_2 \circ \text{Num}_{16}$ .
- ▶ einfacher Fall:  $L_A = A^* = Z_2^*$  und  $L_B = B^* = Z_{16}^*$ .
- ▶ Bedeutungsfunktionen:  $\text{sem}_A = \text{Num}_{16}$  und  $\text{sem}_B = \text{Num}_2$
- ▶ Nachrechnen, dass  $\text{Trans}_{2,16}$  eine Übersetzung ist:

$$\begin{aligned}\text{sem}_B(f(w)) &= \text{Num}_2(\text{Trans}_{2,16}(w)) \\ &= \text{Num}_2(\text{Repr}_2(\text{Num}_{16}(w))) \\ &= \text{Num}_{16}(w) \\ &= \text{sem}_A(w)\end{aligned}$$

- ▶ zum Beispiel bei der Übersetzung von Programmen
- ▶ Was ist Sem, wenn man ein Program in Java übersetzt in eine Folge von Bytecodes?
  - ▶ siehe Vorlesungen zu „Semantik von Programmiersprachen“.

- ▶ **Legalität:** Manchmal ist ein bestimmter Zeichensatz vorgeschrieben (z. B. ASCII in Emails)
- ▶ **Lesbarkeit:** Übersetzungen evtl. kürzer und besser lesbar: vergleiche A3 mit 10100011
- ▶ **Verschlüsselung:** nicht verbesserte Lesbarkeit, sondern
  - ▶ am liebsten gar keine Lesbarkeit, jedenfalls für Außenstehende
  - ▶ siehe Vorlesungen über Kryptographie
- ▶ **Kompression:** Übersetzungen können kürzer sein
  - ▶ und zwar *ohne* größeres Alphabet
  - ▶ gleich: Huffman-Codes
- ▶ **Fehlererkennung und Fehlerkorrektur:**
  - ▶ mache Texte durch Übersetzung länger,
  - ▶ so dass man selbst dann, wenn ein korrekter Funktionswert  $f(w)$  „zufällig“ „kaputt“ (Übertragungsfehler) gemacht wird,
  - ▶ u. U. Fehlererkennung oder gar Fehlerkorrektur möglich
  - ▶ siehe Vorlesungen über Codierungstheorie

- ▶ Forderung  $\text{sem}_A(w) = \text{sem}_B(f(w))$  in Spezialfällen kein Problem:
- ▶ z. B. bei Verschlüsselung und manchen Anwendungen von Kompression:
  - ▶ man will vom Übersetzten  $f(x)$  eindeutig zurückkommen können möchte zum ursprünglichen  $x$ .
  - ▶  $f$  ist injektiv.
- ▶ Dann kann man die Bedeutung  $\text{sem}_B$  im wesentlichen *definieren* durch die Festlegung  $\text{sem}_B(f(x)) = \text{sem}_A(x)$ .
  - ▶ Beachte: Dafür ist Injektivität von  $f$  wichtig!
- ▶ Wenn  $f$  injektiv ist, heie eine bersetzung auch **Codierung**.
- ▶ Die  $f(w)$  heien **Codewrter**.
- ▶ Die Menge  $\{f(w) \mid w \in L_A\}$  heit dann auch ein **Code**.

# Wie spezifiziert man eine Übersetzung?

- ▶ Wenn  $L_A$  unendlich ist, kann man nicht alle Möglichkeiten aufzählen ...
- ▶ Auswege:
  - ▶ Homomorphismen
  - ▶ Block-Codierungen

- ▶ Gegeben:
  - ▶ zwei Alphabete  $A$  und  $B$  und
  - ▶ eine Abbildung  $h : A \rightarrow B^*$ .
- ▶ Zu  $h$  definiert man Funktion  $h^{**} : A^* \rightarrow B^*$  vermöge

$$h^{**}(\varepsilon) = \varepsilon$$

$$\forall w \in A^* : \forall x \in A : h^{**}(wx) = h^{**}(w)h(x)$$

- ▶ Beispiel:
  - ▶  $h(\mathbf{a}) = 10$  und  $h(\mathbf{b}) = 001$
  - ▶  $h^{**}(\mathbf{bab}) = h(\mathbf{b}) \cdot h(\mathbf{a}) \cdot h(\mathbf{b}) = 001 \cdot 10 \cdot 001 = 00110001$
- ▶ Eine solche Abbildung  $h^{**}$  nennt man einen **Homomorphismus**.
- ▶ Homomorphismus heißt  **$\varepsilon$ -frei**, wenn  $\forall x \in A : h(x) \neq \varepsilon$ .

- ▶ Gegeben: Abbildung  $h : A \rightarrow B^*$
- ▶ Frage: ist Abbildung  $h^{**} : A^* \rightarrow B^*$  eine Codierung, also injektiv?
- ▶ im allgemeinen nicht ganz einfach zu sehen.
- ▶ einfacher Spezialfall:  $h$  ist **präfixfrei**
- ▶ Das bedeutet: für *keine zwei verschiedenen* Symbole  $x_1, x_2 \in A$  gilt:  $h(x_1)$  ist ein Präfix von  $h(x_2)$ .

- ▶ allgemeines Problem: nicht alle Wörter aus  $B^*$  sind Codewort, d. h.  $h^{**}$  ist im allgemeinen nicht surjektiv
- ▶ damit Decodierung trotzdem totale Abbildung  $u$  sein kann, definiere  $u : B^* \rightarrow (A \cup \{\perp\})^*$ .
  - ▶ wenn ein  $w \in B^*$  gar nicht decodiert werden kann, dann soll in  $u(w)$  das Symbol  $\perp$  vorkommen

## Beispiel

- ▶ Homomorphismus  $h : \{a, b, c\}^* \rightarrow \{0, 1\}^*$  mit  $h(a) = 1$ ,  $h(b) = 01$ ,  $h(c) = 001$ .
  - ▶ Dieser Homomorphismus ist präfixfrei.
- ▶ dann  $u : \{0, 1\}^* \rightarrow \{a, b, c, \perp\}^*$
- ▶ z. B. soll gelten
  - ▶  $u(\varepsilon) = \varepsilon$
  - ▶  $u(001) = c$
  - ▶  $u(0101) = bb$
  - ▶  $u(0) = \perp$  o. ä.

- ▶ wir schreiben mal hin (und diskutieren das anschließend):

$$u(w) = \begin{cases} \varepsilon & \text{falls } w = \varepsilon \\ au(w') & \text{falls } w = 1w' \\ bu(w') & \text{falls } w = 01w' \\ cu(w') & \text{falls } w = 001w' \\ \perp & \text{sonst} \end{cases}$$

- ▶ sei  $w = 100101 = h(acb)$ ; wir rechnen:

$$\begin{aligned} u(100101) &= au(00101) \\ &= acu(01) \\ &= acbu(\varepsilon) \\ &= acb \end{aligned}$$

$$\text{▶ } u(w) = \begin{cases} \varepsilon & \text{falls } w = \varepsilon \\ au(w') & \text{falls } w = 1w' \\ bu(w') & \text{falls } w = 01w' \\ cu(w') & \text{falls } w = 001w' \\ \perp & \text{sonst} \end{cases}$$

- ▶  $u(100101) = au(00101) = acu(01) = acbu(\varepsilon) = acb$
- ▶ Warum hat das geklappt?
- ▶ In jedem Schritt war klar, welche Zeile der Definition von  $u$  anzuwenden war.
- ▶ denn ...

- ▶ Man spricht hier von **Wohldefiniertheit**
- ▶ immer ein Problem, wenn Funktionswert potenziell „auf mehreren Wegen“ festgelegt;  
dann klar machen:
  - ▶ entweder nur ein Weg „gangbar“
  - ▶ oder auf allen Wegen gleicher Funktionswert
- ▶ einen *präfixfreien* Code kann man also so decodieren:

$$u(w) = \begin{cases} \varepsilon & \text{falls } w = \varepsilon \\ x u(w') & \text{falls } w = h(x)w' \text{ für ein } x \in A \\ \perp & \text{sonst} \end{cases}$$

- ▶ Beachte: das heißt nur, dass man präfixfreie Codes „so einfach“ decodieren kann.

- ▶ Codierung einzelner Zeichen: gleich
- ▶ Wörter werden zeichenweise codiert.
- ▶ UTF-8 ist präfixfrei

Char. number range (hexadecimal)	UTF-8 octet sequence (binary)
0000 0000 - 0000 007F	0xxxxxxx
0000 0080 - 0000 07FF	110xxxxx 10xxxxxx
0000 0800 - 0000 FFFF	1110xxxx 10xxxxxx 10xxxxxx
0001 0000 - 0010 FFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx

- ▶ Determine the number of octets required ...
- ▶ Prepare the high-order bits of the octets ...
- ▶ Fill in the bits marked x ...
  - ▶ Start by putting the lowest-order bit of the character number in the lowest-order position of the last octet of the sequence,
  - ▶ then put the next higher-order bit of the character number in the next higher-order position of that octet, ...
  - ▶ When the x bits of the last octet are filled in, move on to the next to last octet, then to the preceding one, etc. ...

- ▶ Integralzeichen  $\int$  hat Unicode Codepoint `0x222B`
- ▶ `0x222B` in Bits `0010 0010 0010 1011`
- ▶ benutze also die Zeile

---

Char. number range (hexadecimal)	UTF-8 octet sequence (binary)
<code>0000 0800 - 0000 FFFF</code>	<code>1110xxxx 10xxxxxx 10xxxxxx</code>

---

- ▶ also `0010 0010 0010 1011` = `0010 001000 101011`
- ▶ also UTF-8 Codierung `11100010 10001000 10101011`
  
- ▶ Man überlege sich: UTF-8 ist präfixfrei

## Das sollten Sie mitnehmen:

- ▶ Übersetzungen sind in verschiedenen Situationen nützlich
- ▶ Homomorphismen
- ▶ Codes
- ▶ UTF-8

## Das sollten Sie üben:

- ▶ Homomorphismen anwenden
- ▶ Zeichen in UTF-8 codieren

## Codierungen

Von Wörtern zu Zahlen und zurück

Dezimaldarstellung von Zahlen

Andere Zahldarstellungen

Von Zahlen zu ihren Darstellungen

Von einem Alphabet zum anderen

Ein Beispiel: Übersetzung von Zahldarstellungen

Homomorphismen

Beispiel Unicode: UTF-8

## Huffman-Codierung

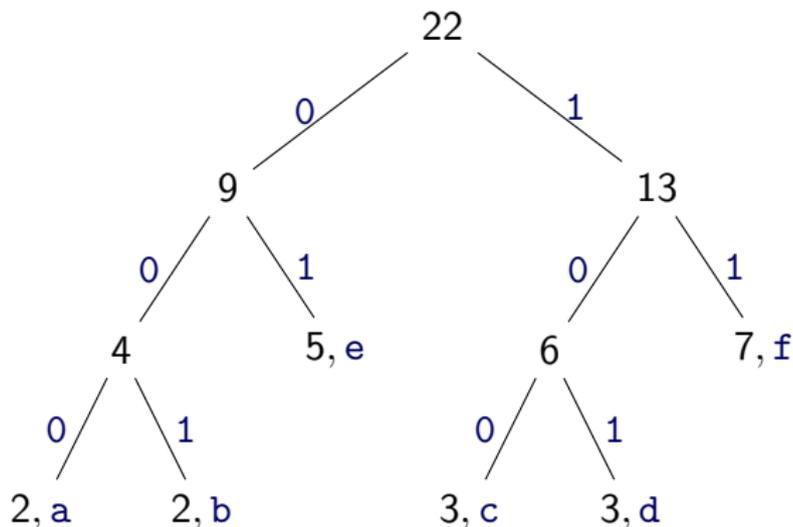
Algorithmus zur Berechnung von Huffman-Codes

Weiteres zu Huffman-Codes

- ▶ Gegeben: ein Alphabet  $A$  und ein Wort  $w \in A^*$
- ▶ Eine sogenannte **Huffman-Codierung** von  $w$  ist
  - ▶ eine Abbildung  $h : A^* \rightarrow Z_2^*$ ,
  - ▶ die ein  $\varepsilon$ -freier Homomorphismus ist.
  - ▶  $h$  also eindeutig festgelegt durch die  $h(x)$  für alle  $x \in A$ .
- ▶ Verwendung z. B. in Kompressionsverfahren wie gzip oder bzip2
- ▶ denn bei Huffman-Codierungen
  - ▶ häufigere Symbole durch kürzere Wörter codiert und
  - ▶ seltenere Symbole durch längere

- ▶ Gegeben
  - ▶ ein  $w \in A^*$  und
  - ▶ die Anzahlen  $N_x(w)$  aller Symbole  $x \in A$  in  $w$
  - ▶ o. B. d. A. alle  $N_x(w) > 0$  (den Rest muss man nicht codieren)
- ▶ Algorithmus zur Bestimmung eines Huffman-Codes arbeitet in zwei Phasen:
  1. es wird ein Baum konstruiert
    - ▶ dessen Blätter den  $x \in A$  entsprechen und
    - ▶ dessen Kanten mit **0** und **1** beschriftet sind
  2. Ablesen der Codes aus dem Baum (Pfadbeschriftungen)

- ▶ Beispiel:  $w = \text{afefbecaffdeddccefbeff}$
- ▶ Baum am Ende:



- ▶ Homomorphismus (präfixfrei!)

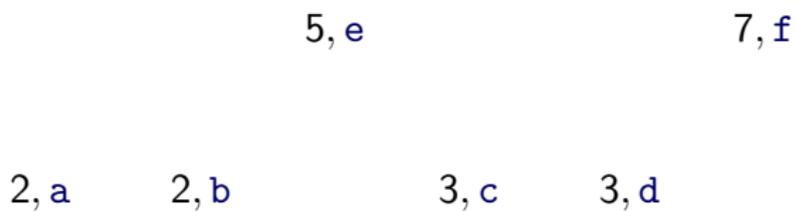
x	a	b	c	d	e	f
h(x)	000	001	100	101	01	11

- ▶ Zu jedem Zeitpunkt hat man
  - ▶ eine Menge  $M_i$  von „noch zu betrachtenden Symbolmengen mit ihren Häufigkeiten“
  - ▶ und eine ebenso große Menge von schon konstruierten Teilbäumen
- ▶ Initialisierung:
  - ▶  $M_0$  ist die Menge aller  $\{(N_x(w), \{x\})\}$  für  $x \in A$ ,
  - ▶ Als Anfang für die Konstruktion des Baumes zeichnet man für jedes Symbol einen Knoten mit Markierung  $(N_x(w), \{x\})$ .
- ▶ Beispiel

$$M_0 = \{ (2, \{a\}) , (2, \{b\}) , (3, \{c\}) , (3, \{d\}) , (5, \{e\}) , (7, \{f\}) \}$$

# Konstruktion des Huffman-Baumes (2)

Anfang im Beispiel:



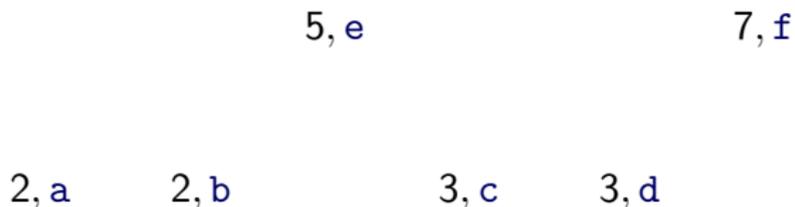
Iterationsschritt des Algorithmus:

- ▶ Solange Menge  $M_i$  noch mindestens zwei Paare enthält, Bestimme Menge  $M_{i+1}$  wie folgt:
  - ▶ Wähle zwei Paare  $(k_1, X_1)$  und  $(k_2, X_2)$ , deren Häufigkeiten zu den kleinsten noch vorkommenden gehören.
  - ▶ Entferne diese Paare aus  $M_i$  und fügt statt dessen das eine Paar  $(k_1 + k_2, X_1 \cup X_2)$  hinzu. Das ergibt  $M_{i+1}$ .
- ▶ im Graphen
  - ▶ Füge einen weiteren Knoten hinzu,
  - ▶ markiert mit der Häufigkeit  $k_1 + k_2$  und Kanten zu den Knoten, die für  $(k_1, X_1)$  und  $(k_2, X_2)$  eingefügt worden waren.

## Konstruktion des Huffman-Baumes (4)

Beispiel:

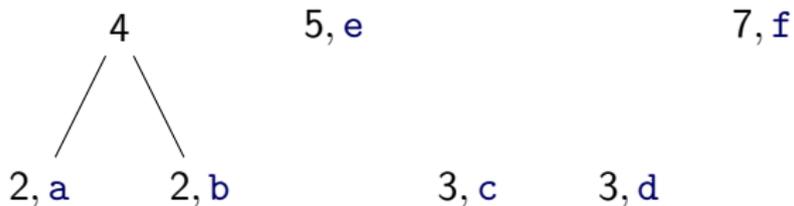
$$M_0 = \{ (2, \{a\}), (2, \{b\}), (3, \{c\}), (3, \{d\}), (5, \{e\}), (7, \{f\}) \}$$



# Konstruktion des Huffman-Baumes (4)

Beispiel:

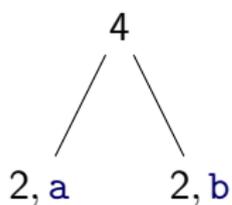
$$M_1 = \{ (4, \{a, b\}), (3, \{c\}), (3, \{d\}), (5, \{e\}), (7, \{f\}) \}$$



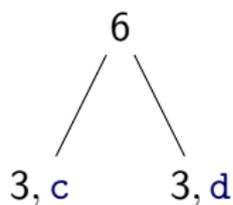
# Konstruktion des Huffman-Baumes (5)

Beispiel:

$$M_2 = \{ (4, \{a, b\}), (6, \{c, d\}), (5, \{e\}), (7, \{f\}) \}$$



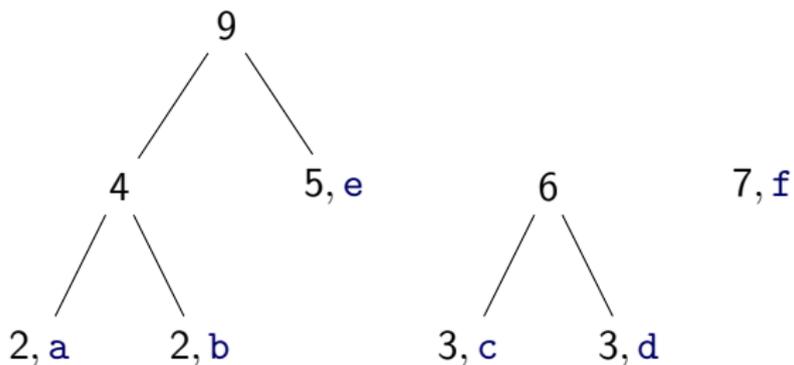
5, e



7, f

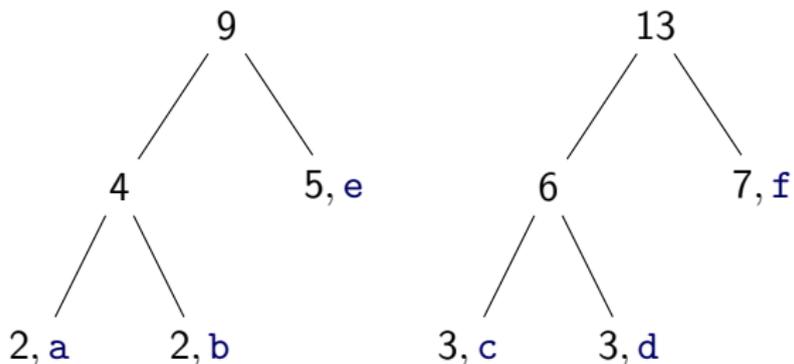
## Beispiel

$$M_3 = \{ (9, \{a, b, e\}), (6, \{c, d\}), (7, \{f\}) \}$$



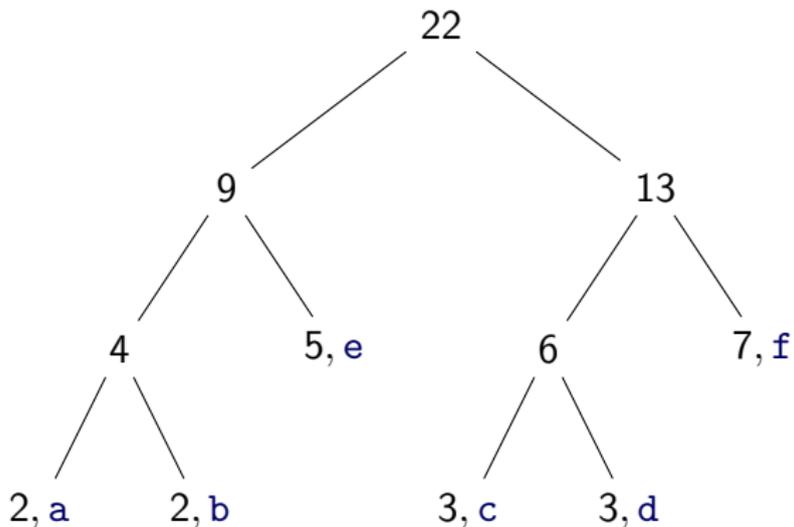
Beispiel

$$M_4 = \{ (9, \{a, b, e\}), (13, \{c, d, f\}) \}$$

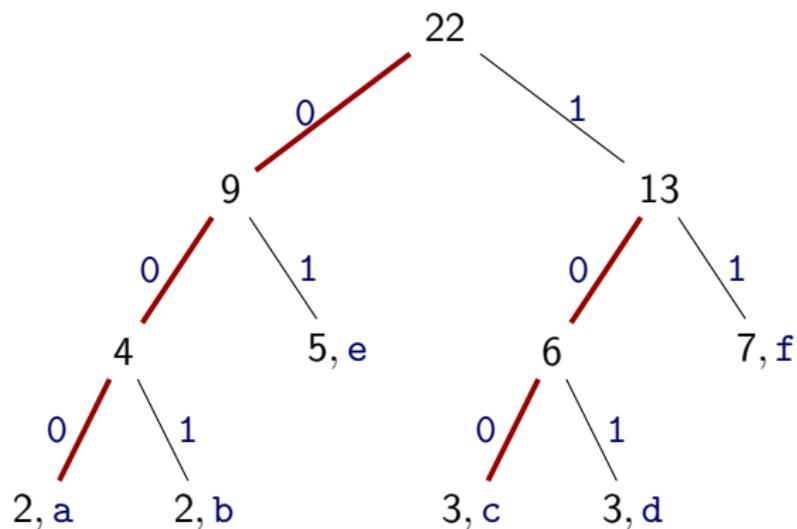


Beispiel

$$M_5 = \{ (22, \{a, b, c, d, e, f\}) \}$$

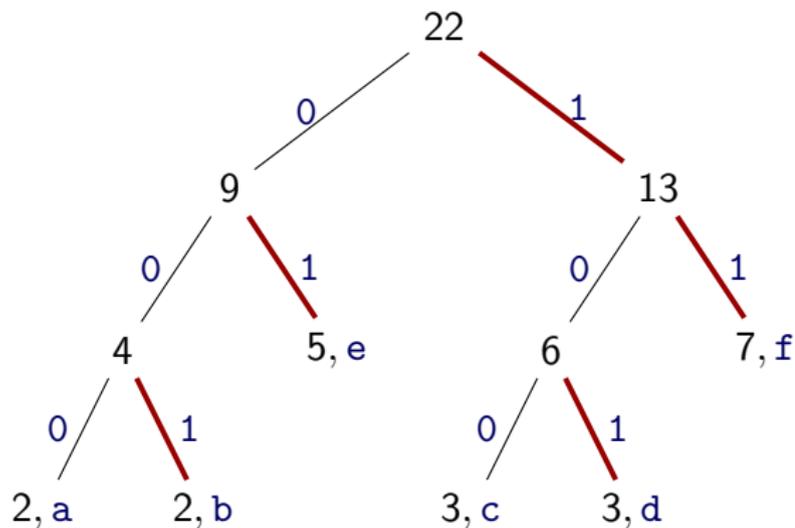


- ▶ nach links führende Kanten werden mit 0 beschriftet
- ▶ nach rechts führende Kanten werden mit 1 beschriftet



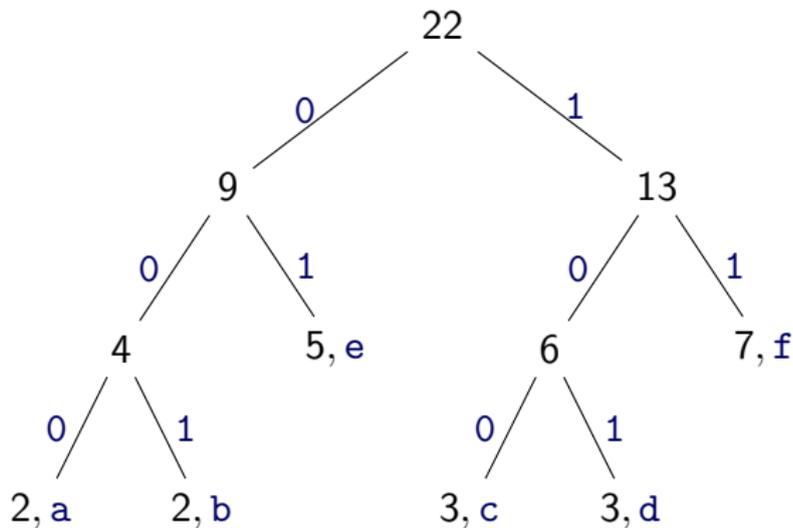
# Beschriftung der Kanten

- ▶ nach links führende Kanten werden mit 0 beschriftet
- ▶ nach rechts führende Kanten werden mit 1 beschriftet



- ▶ gehe auf kürzestem Weg von der Wurzel des Baumes zu dem Blatt, das  $x$  entspricht,
- ▶ konkateniere der Reihe nach alle Symbole, mit denen die Kanten auf diesem Weg beschriftet sind.

- ▶ Beispiel:  $w = \text{afebfecaffdeddccefbeff}$
- ▶ Baum am Ende:



- ▶ Homomorphismus (präfixfrei!)

x	a	b	c	d	e	f
h(x)	000	001	100	101	01	11

- ▶ Huffman-Code nicht eindeutig
  - ▶ im allgemeinen mehrere Möglichkeiten, welche zwei Knoten zu einem neuen vereinigt werden
  - ▶ im Baum links und rechts vertauschbar
  
- ▶ aber alle sind „gleich gut“:
  - ▶ Unter allen präfixfreien Codes führen Huffman-Codes zu kürzesten Codierungen  
*des Wortes, für das die Huffman-Codierung konstruiert wurde.*

- ▶ Verallgemeinerung des obigen Verfahrens:
  - ▶ Betrachte nicht Häufigkeiten einzelner Symbole,
  - ▶ sondern für Teilwörter einer festen Länge  $b > 1$ .
  - ▶ einziger Unterschied: an den Blättern des Huffman-Baumes stehen Wörter der Länge  $b$ .
- ▶ So etwas nennt man eine **Block-Codierung**.
  - ▶ Statt  $h(x)$  für  $x \in A$  festzulegen,
  - ▶ legt man  $h(w)$  für alle **Blöcke**  $w \in A^b$  fest, und
  - ▶ erweitert dies zu einer Funktion  $h : (A^b)^* \rightarrow B^*$ .

## Das sollten Sie mitnehmen:

- ▶ Huffman-Codierung liefert kürzest mögliche präfixfreie Codes
- ▶ „Algorithmus“ zur Bestimmung des Huffman-Baumes
- ▶ warum die Anführungszeichen?

## Das sollten Sie üben:

- ▶ Huffman-Codes berechnen