6 ÜBERSETZUNGEN UND CODIERUNGEN

6.1 von wörtern zu zahlen und zurück

6.1.1 Dezimaldarstellung von Zahlen

Num_{10}

$$\mathrm{Num}_{10}(\varepsilon) = 0 \tag{6.1}$$

$$\text{für jedes } w \in Z_{10}^* \text{ für jedes } x \in Z_{10} : \mathrm{Num}_{10}(wx) = 10 \cdot \mathrm{Num}_{10}(w) + \mathrm{num}_{10}(x) \tag{6.2}$$

- noch ein Beispiel oder langweilig?
- Beweis, dass Definition sinnvoll:

6.1 Lemma. Durch die Gleichungen 6.1 und 6.2 ist Num_{10} wohldefiniert, das heißt, für jedes Wort $w \in Z^*$ wird eindeutig der Funktionswert $\mathrm{Num}_{10}(w)$ festgelgt.

Um das mittels vollständiger Induktion zu beweisen, formulieren wir etwas um:

6.2 Lemma. Durch die Gleichungen 6.1 und 6.2 ist Num_{10} wohldefiniert, das heißt,
für jede Wortlänge $n \in \mathbb{N}_0$ gilt: für alle $w \in Z^n$ wird eindeutig der Funktionswert $\mathrm{Num}_{10}(w)$ festgelgt.

6.3 Beweis. Beweis durch Induktion über die Wortlänge n:

Induktionsanfang $n = 0$: In diesem Fall ist zu beweisen: Für jedes Wort w der Länge $n = 0$ ist $\mathrm{Num}_{10}(w)$ festgelegt, und zwar eindeutig.

Das ist leicht: Wenn $|w| = 0$ ist, dann ist $w = \varepsilon$. Tatsächlich legt Gleichung (6.1) offensichtlich eindeutig einen Funktionswert $\mathrm{Num}_{10}(\varepsilon)$ fest, nämlich 0. Und Gleichung (6.2) legt *keinen* Funktionswert für $\mathrm{Num}_{10}(\varepsilon)$ fest, denn die auf der linken Seite auftretenden Argumente haben alle eine Länge $|wx| = |w| + |x| = |w| + 1 \geq 1$, sind also ganz bestimmt *nicht* das leere Wort.

Induktionsvoraussetzung: für ein beliebiges aber festes $n \in \mathbb{N}_0$ gelte: Für alle Wörter $w \in Z^n$ ist $\mathrm{Num}_{10}(w)$ eindeutig definiert.

Induktionsschritt $n \rightsquigarrow n + 1$: Nun müssen wir beweisen, dass die Richtigkeit der Aussage des Lemmas für ein n zwingend auch die Richtigkeit der Aussage für $n + 1$ nach sich zieht.

Bezeichne w' ein beliebiges Wort der Länge $n + 1$. Wir müssen zeigen: $\mathrm{Num}_{10}(w')$ ist eindeutig festgelegt. Das geht so:

Wenn $|w'| = n + 1$, dann enthält w' mindestens ein Symbol, also auch ein letztes. Bezeichnen wir das mit x. Dann ist w' von der Form $w' = wx$ mit $w \in Z^n$ und $x \in Z$.

In der Definition von Num_{10} passt *nur* Gleichung 6.2:

$$\mathrm{Num}_{10}(w') = \mathrm{Num}_{10}(wx) = 10 \cdot \mathrm{Num}_{10}(w) + \mathrm{num}_{10}(x)$$

Nach IV ist $\mathrm{Num}_{10}(w)$ eindeutig festgelegt, also auch $\mathrm{Num}_{10}(w')$.

■

6.1.2 Andere unbeschränkte Zahldarstellungen

Beispielrechungen

- klar machen, wie allgemein bei Basis k die Umwandlung funktioniert: $\mathrm{Num}_k(wx) = k \cdot \mathrm{Num}_k(w) + \mathrm{num}_k(x)$.
- Beispiel rechnen, z.B. $\mathrm{Num}_3(111) = \cdots = 13$.
- $\mathrm{Num}_2(1) = 1$, $\mathrm{Num}_2(11) = 3$, $\mathrm{Num}_2(111) = 7$, $\mathrm{Num}_2(1111) = 15$,
  Wer sieht allgemein: für jedes $m \in \mathbb{N}_0 : \mathrm{Num}_2(1^m) = 2^m - 1$?
  Wie überträgt sich das auf den Fall $k = 3$? für jedes $m \in \mathbb{N}_0 : \mathrm{Num}_3(2^m) = 3^m - 1$.

6.1.3 Ganzzahlige Division mit Rest

Rechnen mit div und mod

$$x = y \cdot (x \operatorname{\mathbf{div}} y) + (x \operatorname{\mathbf{mod}} y) \qquad \text{und} \qquad 0 \le (x \operatorname{\mathbf{mod}} y) < y$$

- Bitte das Rechnen mit **div** und **mod** üben.
  Vielleicht mal einfach eine Tabelle ausfüllen (lassen) der Form

x	0	1	2	3	4	5	6	7	8	9	10	11
x **div** 4	0	0	0	0	1	1	1	1	2	2	2	2
4(x **div** 4)	0	0	0	0	4	4	4	4	8	8	8	8
x **mod** 4	0	1	2	3	0	1	2	3	0	1	2	3

- Klar machen, dass x **mod** 2 was mit gerade und ungerade zu tun hat.

6.1.4 Von Zahlen zu ihren Darstellungen

6.1.5 Beschränkte Zahlbereiche und Zahldarstellungen

- Die Abkürzung $\mathbb{K}_\ell$ ist nicht allgemein üblich, aber ich wollte eine griffige Abkürzung.
- Man berechne Zweierkomplementdarstellungen; man nehme Länge 5
  - man lasse überlegen, dass $\mathbb{K}_5 = \{-16, \dots, -1, 0, 1, \dots, 15\}$
  - man lasse z. B. berechnen
    - $\mathrm{Zkpl}_6(0) = 00000$
    - $\mathrm{Zkpl}_6(1) = 00001$
    - $\mathrm{Zkpl}_6(2) = 00010$
    - $\mathrm{Zkpl}_6(15) = 01111$
    - $\mathrm{Zkpl}_6(-1) = 11111$
    - $\mathrm{Zkpl}_6(-2) = 11110$
    - $\mathrm{Zkpl}_6(-15) = 10001$
    - $\mathrm{Zkpl}_6(-16) = 10000$

6.2 von einem alphabet zum anderen

Übersetzungen

Warum macht man Übersetzungen? Fällt jemandem noch was ein außer

- Lesbarkeit
- Kompression
- Verschlüsselung
- Fehlererkennung und Fehlerkorrektur

6.2.1 Ein Beispiel: Übersetzung von Zahldarstellungen

6.2.2 Homomorphismen

Homomorphismen

Bitte beachten: Seit Wintersemester 2015/2016 werden Homomorphismen etwas anders eingeführt. Ein Homomorphismus ist eine Abbildung $h : A^* \to B^*$ mit $h(w_1 w_2) = h(w_1)h(w_2)$. Bei Homomorphismen gibt es kein h^{**} mehr.

Was es noch gibt ist, dass man aus einer Abbildung $f : A \to B^*$ (Def.bereich ist nur A, nicht A^*) eine Abbildung $f^{**} : A^* \to B^*$ macht.

- Beispiel:
  - $h(a) = \mathtt{001}$ und $h(b) = \mathtt{1101}$
  - dann ist $h(\mathtt{bba}) = h(\mathtt{b})h(\mathtt{b})h(\mathtt{a}) = \mathtt{1101} \cdot \mathtt{1101} \cdot \mathtt{001} = \mathtt{11011101001}$
- ε-freier Homomorphismus: Warum will man das? Sonst geht „Information verloren“. keine Codierung mehr; Betrachte
  - $h(a) = \mathtt{001}$ und $h(b) = \varepsilon$
  - angenommen $h(w) = \mathtt{001}$ Was war dann w? Man weiß nur: es kam genau ein $\mathtt{a}$ vor, aber wieviele $\mathtt{b}$ und wo ist nicht klar.
- Information kann aber auch anders verloren gehen;
  - z. B. $h(\mathtt{a}) = \mathtt{0}$, $h(\mathtt{b}) = \mathtt{1}$, $h(\mathtt{c}) = \mathtt{10}$ oder
  - $h(\mathtt{a}) = h(\mathtt{b})$, oder ...

  allgemeine Formalisierung von „Information geht verloren“ suchen lassen: es gibt Wörter $w_1 \neq w_2$ (verschiedene!) mit $h(w_1) = h(w_2)$
- präfixfreier Code: für keine zwei verschiedenen Symbole $x_1, x_2 \in A$ gilt: $h(x_1)$ ist ein Präfix von $h(x_2)$.

  Beispiel
  - $h(\mathtt{a}) = \mathtt{001}$ und $h(\mathtt{b}) = \mathtt{1101}$ ist präfixfrei
  - $h(\mathtt{a}) = \mathtt{01}$ und $h(\mathtt{b}) = \mathtt{011}$ ist nicht präfixfrei
  - Präfixfreiheit leicht zu sehen, wenn alle $h(x)$ gleich lang sind: präfixfrei $\Longleftrightarrow$ injektiv; Beispiel: ASCII

6.2.3 Beispiel Unicode: UTF-8 Codierung

- Man könnte, wenn die Zeit reicht, ja mal für ein paar Zeichen die UTF-8 Codierung be stimmen. Zum Beispiel gibt es für π in Unicode ein Zeichen, nämlich das mit der Nummer `0x03C0`.

  Wenn ich den Algorithmus richtig gemacht habe, ergibt sich

  - Code Point `0x03C0`
  - in Bits `0000 0011 1100 0000` = `00000 01111 000000`
  - man benutzt die Zeile

Char. number range (hexadecimal)	UTF-8 octet sequence (binary)
`0000 0080` - `0000 07FF`	`110`$xxxxx$ `10`$xxxxxx$

  - also UTF-8 Codierung `11001111 10000000`
- Man mache sich klar, dass UTF-8 präfixfrei ist.

6.3 Huffman-Codierung

6.3.1 Algorithmus zur Berechnung von Huffman-Codes

Nehmen Sie acht Symbole: `a`, `b`, `c`, `d`, `e`, `f`, `g`, `h`

- 1. Fall: Jedes Zeichen kommt genau einmal vor.

  Huffman-Code-Baum erstellen, Wort `badcfehg` codieren, wie lang wird die Codierung?
- 2. Fall: `a` kommt einmal vor, `b` zweimal, `c` 4-mal, `d` 8-mal, `e` 16-mal, `f` 32-mal, `g` 64-mal, `h` 128-mal.

  Huffman-Code-Baum erstellen,

  Ergebnis z. B.

x	a	b	c	d	e	f	g	h
h(x)	`0000000`	`0000001`	`000001`	`00001`	`0001`	`001`	`01`	`1`

  aber nicht das Wort `abbcccc`...$\texttt{h}^{128}$ codieren, sondern das Wort `badcfehg`; wie lang wird die Codierung? ... über 4 mal so lang
- Wie lange wird ein Wort mit zweiter Zeichenverteilung, wenn man es mit dem ersten Code codiert?

  Dreimal so lang, weil jeder Buchstabe durch 3 Bits codiert wird.
- Wie lange wird ein Wort mit erster Zeichenverteilung, wenn man es mit dem zweiten Code codiert?

  Ziel: Sehen, dass Huffman-Codierung irgendwie zu funktionieren scheint.

6.3.2 Weiteres zu Huffman-Codes

Blockcodierung mit Huffman

- Verallgemeinerung: nicht von den Häufigkeiten einzelner Symbole ausgehen, sondern für Teilwörter einer festen Länge $b > 1$ die Häufigkeiten berechnen.
- Beispiel: Man hat einen Text über dem Alphabet $\{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$, der nur aus Teilwörtern der Länge 10 zusammengesetzt ist, die denen in jedem immer nur ein Symbol vorkommt, also z. B. `aaaaaaaaaabbbbbbbbbbddddddddddcccccccccc` usw. Angenommen $\mathtt{a}^{10}, \dots, \mathtt{d}^{10}$ kommen alle gleich häufig vor. Wie lang ist dann die Huffman-Codierung? Ein Fünftel, weil jeder Zehnerblock durch zwei Bits codiert wird.