Tutorium 17, #4

Max Göckel– *uzkns@student.kit.edu*

Institut für Theoretische Informatik - Grundbegriffe der Informatik

Num_k

Wenn w ein Wort ist, das eine Zahl zur Basis k darstellt, so ist $Num_k(w)$ die Darstellung dieses Wortes/dieser Zahl im Dezimalsystem.

- Definition der Funktion mit w=w'·x.
  - $Num_k(\epsilon) = 0$
  - $Num_k(w'x) = k \cdot Num_k(w') + num_k(x)$ mit $x \in \mathbb{Z}_k$ und $w' \in \mathbb{Z}_k^*$
  - $num_k(x) = x$

Num_k: Beispiel

Sei $w_1 = 5_8$ (5 zur Basis 8).

$$
\begin{aligned}
&Num_8(5)\\
&= 8 \cdot Num_8(\epsilon) + num_8(5)\\
&= 8 \cdot 0 + 5\\
&= 5.
\end{aligned}
$$

Num_k: **Aufgabe**

Sei w_2=B66 in der Basis 16

- Gib die Repräsentation der Zahl im Dezimalsystem mithilfe der Num_k-Funktion an.

Num_k: Lösung

Sei $w_3 = B66_{16}$ (In Hexadezimal gilt B = 11, also $w = 11 \cdot 6 \cdot 6$ in der Basis 16).

$Num_{16}(B66)$
$= 16 \cdot Num_{16}(B6) + num_{16}(6)$
$= 16 \cdot (16 \cdot Num_{16}(\epsilon B) + num_{16}(6)) + num_{16}(6)$
$= 16 \cdot (16 \cdot (16 \cdot Num_{16}(\epsilon) num_{16}(B)) + num_{16}(6)) + num_{16}(6)$
$= 16 \cdot (16 \cdot 11 + 6) + 6$
$= (16 \cdot 16 \cdot 11) + (16 \cdot 6) + 6$
$= 2816 + 96 + 6$
$= 2918$

$Repr_k$

$Repr_k$ ist die Darstellung einer Dezimalzahl in der Basis k.

- Definition:
  - $n < k$: $repr_k(n)$
  - $n \geq k$: $Repr_k(n \text{ div } k) \cdot repr_k(n \text{ mod } k)$

$Repr_k$: Beispiel

$w = 29$ und $k = 3$.

$$
\begin{aligned}
& Repr_3(29) \\
&= Repr_3(\tfrac{29}{3}) \cdot repr_3(29 \bmod 3) \\
&= Repr_3(9) \cdot 2 \\
&= Repr_3(\tfrac{9}{3}) \cdot repr_3(9 \bmod 3) \cdot 2 \\
&= Repr_3(3) \cdot 0 \cdot 2 \\
&= Repr_3(1) \cdot repr_3(0) \cdot 0 \cdot 2 \\
&= repr_3(1) \cdot 0 \cdot 0 \cdot 2 \\
&= 1 \cdot 0 \cdot 0 \cdot 2 = 1002_3
\end{aligned}
$$

$Repr_k$: Aufgabe

w=53 und k=5.

- Stelle w mithilfe der $Repr_k$-Funktion dar.

Fakultät für Informatik
Institut für Theoretische Informatik

$Repr_k$: Lösung

$w = 53$ und $k = 5$.

$$
\begin{aligned}
& Repr_5(53) \\
&= Repr_5\left(\tfrac{53}{5}\right) \cdot repr_5(53 \bmod 5) \\
&= Repr_5(10) \cdot repr_5(3) \\
&= Repr_5\left(\tfrac{10}{5}\right) \cdot repr_5(10 \bmod 5) \cdot 3 \\
&= Repr_5(2) \cdot 0 \cdot 3 \\
&= 2 \cdot 0 \cdot 3 = 203_5
\end{aligned}
$$

Binärsystem

Im Dezimalsystem gibt es die Ziffern 1..9.

Im Binär- oder Dualsystem gibt es nur 2 Ziffern: $1, 2$.
Die Wertigkeit jeder Ziffer Z ist dabei $Z \cdot 2^i$.

Binärtabelle

Wertigkeit:	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0

Zahl 1100: $1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 8 + 4 + 0 + 0 = 12$

Zweierkomplement

Zweierkomplementdarstellung ($\mathbb{K}_n$) ist die übliche Darstellung von ganzen Zahlen im PC. Dabei "opfern" wir das erste Binär-Bit für das Vorzeichen.
Zahlen werden gleichmäßig auf den positiven und negativen Bereich verteilt. Mit der 0 ist im positiven Bereich eine Zahl weniger als im negativen.

Definition

- $\mathbb{K}_n = \{x \in \mathbb{Z} | -2^{n-1} \leq x \geq 2^{n-1} - 1\}$

- $\mathbb{K}_7 = \{-64, -63, ..., ..., 62, 63\}$ (Alle Zahlen bis 2^6, im positiven eine Zahl weniger)

$Zkpl_k$

Umrechung eriner n-stelligen Binärzahl ins Zweierkomplement mit $Zkpl_n$.

Vorgehen bei negativen Zahlen:

- Nullen durch Einsen erstzen und Einsen durch Nullen (Invertieren)
- 1 dazu addieren
- Schauen ob die Zahl vorher positiv oder negativ war, entsprechend das erste Bit anpassen

Bei positiven Zahlen:

- Nichts tun, evtl. auf richtige Länge mit 0'len auffüllen

$Zkpl_k$: Beispiel

Beispielzahl ist -01101011_2

Vorgehen:

- Invertieren: 10010100
- +1: 10010101

$Zkpl_k$: Aufgaben

Rechnet die fogenden Zahlen mit Zwischenschritten um:

- 5 zuerst in binär, denn in ZK_4
- -100101_2 in ZK_9
- 11110_{ZK} zurück in binär

$Zkpl_k$: Lösung

Rechnet die fogenden Zahlen mit Zwischenschritten um:

- $5 = 101_2 = 0101_{ZK}$
- $-100101_2 = 111011011_{ZK}$
- $11110_{ZK} = -10_2$

Homomorphismen

A,B so bildet eine Abbildung $h : A \to B^*$ einen Buchstaben von A uf ein Wort aus B ab.
ein Homorphismus $h^{**} : A^* \to B^*$ macht das selbe mit einem ganzen Wort aus A, wobei jedes Zeichen einzeln abgebildet wird.
Ist ein Homomorphismus präfixfrei so existiert eine Umkehrfunktion die aus dem Wort aus B* wider das Wort in A* macht

Definition

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

Homomorphismen: Beispiel

Sei $h(a) = 101, H(b) = 0, h(c) = 1$

- $h^{**}(a) = h(a) = 101$
- $h^{**}(cbc) = h(c) \cdot h(b) \cdot h(c) = 101$

Huffman-Codierung

Die Huffman-Codierung ist ein präfixfreier Homomorphismus $A^* \to \mathbb{Z}_2{}^*$, der häufigen Zeichen eine möglichst kurze Abbildung gibt.

Sei $N_w(x)$ die Anzahl der Vorkommnisse des Buchstaben x in w und "$N_w(x), x$" die Beschriftung des Blattes in einem Baum

- Verbinde die zwei "kleinsten" Werte
- Schreibe $N_w(x_1) + N_w(x_2)$ in den neuen Knoten
- So lange wiederholen, bis der Baum komplett ist

Huffman: Beispiel

Karlsruher Institut für Technologie

w = bcccabdd, so sind $N_w(a) = 1$, $N_w(b) = 2$, $N_w(c) = 3$, $N_w(d) = 2$ die Häufigkeiten in w.

Huffman: Aufgabe

Stelle das folgende Wort in einem Huffman-Baum dar und erstelle das Codewort:

w = ededddbcfbedaccb

Huffman: Frage

Ist ein Huffman-Baum eindeutig?

-

Kann man ein Huffman-Wort wieder decodieren?

-

Huffman: Antwort

Ist ein Huffman-Baum eindeutig?

- Nein, da bei mehreren gleich-häufig vorkommenden Wörern man sich aussuchen kann, welche Kante man zuerst verbindet.

Kann man ein Huffman-Wort wieder decodieren?

- Ja, dafür benötigt man aber den Baum.

Huffman-Codierung

Kommen viele Teilwörter in einem Wort w häufig vor, so macht es Sinn die Codierung mit kurzen Wörtern anstelle von einzelnen Buchtaben durchzuführen.
Ansonsten bleibt das Vorgehen gleich, die Suche nach den besten Teilwörter n kann aber etwas Zeit in Anspruch nehmen.

Beispielwort: $w = abcabccbdabc = abc \cdot abc \cdot cbd \cdot abc$

- h(abc) = 0, h(cbd) = 1
- $h^{**}(w) = 0010$
- $|w| = 12, |h^{**}(w)| = 4$, Platzeinsparung um den Faktor 3

Fakultät für Informatik
Institut für Theoretische Informatik