Technische Informatik
Zahlendarstellung

Thorsten Thormählen

22. Oktober 2024

Teil 2, Kapitel 1

Dies ist die Druck-Ansicht.

Aktiviere Präsentationsansicht

Steuerungstasten

→ nächste Folie (auch Enter oder Spacebar).
← vorherige Folie
d schaltet das Zeichnen auf Folien ein/aus
p wechselt zwischen Druck- und Präsentationsansicht
CTRL + vergrößert die Folien
CTRL - verkleinert die Folien
CTRL 0 setzt die Größenänderung zurück

Notation

Typ	Schriftart	Beispiele
Variablen (Skalare)	kursiv	a, b, x, y
Funktionen	aufrecht	$\mathrm{f}, \mathrm{g}(x), \max(x)$
Vektoren	fett, Elemente zeilenweise	$\mathbf{a}, \mathbf{b} = \begin{pmatrix} x \\ y \end{pmatrix} = (x, y)^\top,$ $\mathbf{B} = (x, y, z)^\top$
Matrizen	Schreibmaschine	$\mathtt{A}, \mathtt{B} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$
Mengen	kalligrafisch	$\mathcal{A}, B = \{a, b\}, b \in \mathcal{B}$
Zahlenbereiche, Koordinatenräume	doppelt gestrichen	$\mathbb{N}, \mathbb{Z}, \mathbb{R}^2, \mathbb{R}^3$

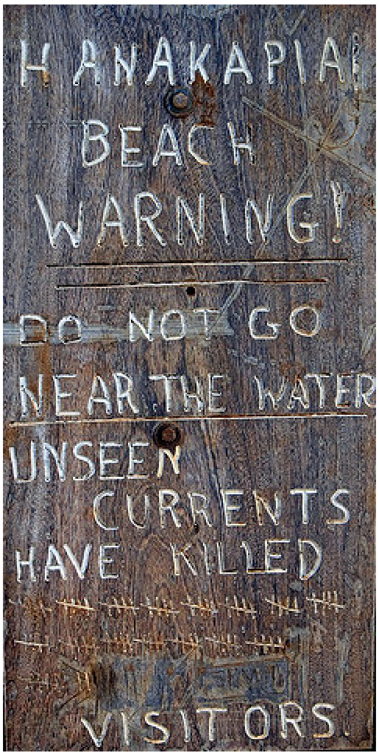
"Es gibt 10 Gruppen von Menschen: Diejenigen, die Binärkodierungen verstehen und die anderen."

[Quelle: Wikipedia]

Inhalt

Zahlendarstellung durch Kerben und Striche

- Eine Zahl x wird durch x-fache Wiederholung eines speziellen Zeichens (Strich, Kerbe, etc.) dargestellt
- Stichsysteme wurden schon sehr früh von den Menschheit verwendet
- Wird heute noch häufig verwendet: z.B. Kaffee-Liste, Inventur
- Zwar praktisch, wenn die Zahlen klein sind, aber bei großen Zahlen nicht mehr anwendbar
- Addition ergibt sich natürlich



[**Bildquelle:** Unseen Currents Have Killed 82 Visitors, by eeetthaannn, Creative Commons License]

Römische Zahlen

Zahlensymbol	Wertigkeit
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Römische Zahlen

Bei den Römischen Zahlen ist die *relative* Position der Zeichen wichtig

Beispiele:

I	= 1	X	= 10
II	= 2	XI	= (10+1) = 11
III	= 3	XII	= (10+2) = 12
IV	= (5-1) = 4	XXXIX	= (30+10-1) = 39
V	= 5	XL	= (50-10) = 40
VI	= (5+1) = 6	L	= 50
VII	= (5+2) = 7	LIX	= (50+10-1)=59
VIII	= (5+3) = 8	LX	= 60
IX	= (10-1) = 9	XC	= (100-10) = 90

Dezimalsystem

Das von uns im Alltag verwendete Dezimalsystem kennt 10 Zahlensymbole: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

Dabei ist die *absolute* Position der Zeichen wichtig, um den Zahlenwert zu ermitteln

Beispiel: $123 \neq 321$

Beispiel: $5124 = 5 \cdot 1000 + 1 \cdot 100 + 2 \cdot 10 + 4$

Der Zahlenwert w berechnet sich also wie folgt aus den Ziffern $z_{n-1}, \ldots, z_1, z_0$ einer Dezimalzahl:

$$w = z_{n-1} \cdot 1\underbrace{0 \ldots 0}_{n-1} + \ldots z_2 \cdot 100 + z_1 \cdot 10 + \; z_0 \cdot 1 =$$

$$= \sum_{i=0}^{n-1} z_i \cdot 10^i$$

Bei dem Dezimalsystem handelt es sich um ein so genanntes *Stellenwertsystem*

Das Dezimalsystem ist ein Stellenwertsystem mit der Basis 10

Stellenwertsysteme: b-adische Darstellung

Sei $b > 1$ eine beliebige natürliche Zahl ($b \in \mathbb{N}$), dann heisst die Menge $\{0, \ldots, b-1\}$ das Alphabet des b-adischen Zahlensystems (mit der Basis b)

- Dezimalsystem:
  $b = 10 \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- Dualsystem (Binärsystem):
  $b = 2 \rightarrow \{0, 1\}$
- Oktalsystem:
  $b = 8 \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7\}$
- Hexadezimalsystem:
  $b = 16 \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f\}$

Stellenwertsysteme: b-adische Darstellung

Wie schon festgestellt, berechnet sich ein Zahlenwert w im Dezimalsystem gemäß:

$$w = \sum_{i=0}^{n-1} z_i \cdot 10^i$$

Dies lässt sich verallgemeinern. Für beliebige Basen $b > 1$, gilt:

$$w = \sum_{i=0}^{n-1} z_i \cdot b^i$$

wobei n die Anzahl der Stellen der Zahl w angibt

Ziffernschreibweise: $w = (z_{n-1} \ldots z_2 z_1 z_0)_b$

Dualsystem (Binärsystem)

Der Basis $b = 2$ mit einem Alphabet von zwei Zahlensymbolen, 0 und 1, kommt in der Informatik eine besondere Bedeutung zu, da sich zwei Symbole leicht elektrisch kodieren lassen:

- "kein Strom fließt" entspricht typischerweise der 0
- "Strom fließt" der 1
- Jede Stelle einer Binärzahl wird als "Bit" ("Binary Digit") bezeichnet
- 1 Bit ist demnach die kleinste Informationsmenge, die gespeichert werden kann
- 8 Bits werden als 1 Byte bezeichnet

Die Bitfolge $(01001101)_2$ entspricht der Dezimalzahl $(77)_{10}$

$$\begin{aligned}
&(01001101)_2 \\
&= 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \\
&= 0 \cdot 128 + 1 \cdot 64 + 0 \cdot 32 + 0 \cdot 16 + 1 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 \\
&= 64 + 8 + 4 + 1 \\
&= 77
\end{aligned}$$

Oktal- und Hexadezimalsystem

Oktalsystem:

Die Oktalzahl $(115)_8$ entspricht der Dezimalzahl $(77)_{10}$

$$\begin{aligned}(115)_8 &= 1 \cdot 8^2 + 1 \cdot 8^1 + 5 \cdot 8^0 \\ &= 1 \cdot 64 + 1 \cdot 8 + 5 \cdot 1 \\ &= 77\end{aligned}$$

Hexadezimalsystem:

Die Zahl $(4D)_{16}$ entspricht der Dezimalzahl $(77)_{10}$

$$\begin{aligned}(4D)_{16} &= 4 \cdot 16^1 + D \cdot 16^0 \\ &= 64 + 13 \\ &= 77\end{aligned}$$

Zahlensysteme mit verschiedenen Basen

Dies ist ein interaktiver Zähler. Durch Klicken auf den Zähler wird die Zahl erhöht:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0 (10)
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0 (2)

+ - Reset

Es können verschiedene Basen ausgewählt werden:

Basis (oben): 10   Basis (unten): 2

Beobachtung: Kleine Basen benötigen zwar ein kleineres Alphabet, aber dafür mehr Stellen

"Es gibt 10 Gruppen von Menschen: Diejenigen die Ternärkodierungen verstehen, jene die es für Binärkodierung halten und die anderen."

[Quelle: Wikipedia]

Quiz

Frage: Die Zahl $(1220)_3$ im Ternärsystem soll in das Dezimalsystem umgewandet werden. Was ist das Ergebnis?

Antwort 1: **51**

Antwort 2: **17**

Antwort 3: **24**

Antwort 4: **49**

Antwort 5: **keine Ahnung**

Am Online-Quiz teilnehmen durch Besuch der Webseite: www.onlineclicker.org

Rationale Zahlen

Eine rationale Zahl $w \in \mathbb{Q}$ kann ebenfalls in der b-adischen Darstellung angegeben werden

Dazu wird der Nachkommaanteil durch negative Exponenten repräsentiert

Beispiel aus dem uns vertrauten Dezimalsystem:

$$913,64 = 9 \cdot 10^2 + 1 \cdot 10^1 + 3 \cdot 10^0 + 6 \cdot 10^{-1} + 4 \cdot 10^{-2}$$

Dies lässt sich wieder verallgemeinern:

$$\begin{aligned} w &= (z_{n-1} \dots z_2 z_1 z_0, z_{-1} \dots z_{-m})_b \\ &= \sum_{i=-m}^{n-1} z_i \cdot b^i \end{aligned}$$

wobei n die Anzahl der Vorkommastellen
und m die Anzahl der Nachkommastellen angibt

Beispiel im Binärsystem:

$$\begin{aligned} (11,101)_2 &= 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} \\ &= 2 + 1 + 0.5 + 0,125 = (3,625)_{10} \end{aligned}$$

Dezimal-Präfixe

Für viele Zehnerpotenzen gibt es standardisierte Präfixe, die vor den eigentlichen physikalischen Einheiten stehen, z.B. 1000 m = 1 km

10^1	10^2	10^3	10^6	10^9	10^{12}	10^{15}	10^{18}	10^{21}	10^{24}
Deka	Hekto	Kilo	Mega	Giga	Tera	Peta	Exa	Zetta	Yotta
da	h	k	M	G	T	P	E	Z	Y

10^{-1}	10^{-2}	10^{-3}	10^{-6}	10^{-9}	10^{-12}	10^{-15}	10^{-18}	10^{-21}	10^{-24}
Dezi	Zenti	Milli	Mikro	Nano	Piko	Femto	Atto	Zepto	Yokto
d	c	m	µ	n	p	f	a	z	y

[Quelle: Wikipedia]

Binär-Präfixe

Wenn in der Informatik von 1 kByte (Kilobyte) gesprochen wird, bezeichnet dies nicht immer 1000 Bytes, sondern häufig 1024 Bytes. Dies führt zu Verwirrungen. Daher sollten für die Potenzen von 1024 besser die folgenden Präfixe laut IEC verwendet werden:

Name	Präfix	Wert
kibi	Ki	$2^{10} = 1024^1 = 1.024$
mebi	Mi	$2^{20} = 1024^2 = 1.048.576$
gibi	Gi	$2^{30} = 1024^3 = 1.073.741.824$
tebi	Ti	$2^{40} = 1024^4 = 1.099.511.627.776$
pebi	Pi	$2^{50} = 1024^5 = 1.125.899.906.842.624$
exbi	Ei	$2^{60} = 1024^6 = 1.152.921.504.606.846.976$
zebi	Zi	$2^{70} = 1024^7 = 1.180.591.620.717.411.303.424$
yobi	Yi	$2^{80} = 1024^8 = 1.208.925.819.614.629.174.706.176$

[Quelle: Wikipedia]

Umwandlung zwischen Zahlensystemen

Zur Entwicklung eines Algorithmus zum Umrechnen zwischen Zahlensystemen betrachten wir was passiert, wenn wir die hergeleitete Formel zur b-adischen Darstellung durch die Basis b dividieren:

$$w = \sum_{i=0}^{n-1} z_i \cdot b^i \Leftrightarrow \frac{w}{b} = \frac{1}{b} \sum_{i=0}^{n-1} z_i \cdot b^i$$
$$= \underbrace{\left(\sum_{i=0}^{n-2} z_{i+1} \cdot b^i \right)}_{s_1} + \frac{z_0}{b}$$

Der erste Summand s_1 ist das ganzzahlige Ergebnis der Division und z_0 der Rest der Division

Damit wurde gezeigt, dass die letzte Ziffer z_0 aus dem Rest der Division mit der gewünschten Basis b errechnet werden kann

Das ganzzahlige Ergebnis der Division stellt den Zahlenwert der verbleibenden Ziffern (ohne z_0) dar. Durch rekursive Anwendung können alle Ziffern bestimmt werden.

Umwandlung zwischen Zahlensystemen

Beispiel: Umrechnung der Dezimalzahl 6485 ins Binärsystem:

```
6485 / 2 = 3242  Rest 1
3242 / 2 = 1621  Rest 0
1621 / 2 =  810  Rest 1
 810 / 2 =  405  Rest 0
 405 / 2 =  202  Rest 1
 202 / 2 =  101  Rest 0
 101 / 2 =   50  Rest 1
  50 / 2 =   25  Rest 0
  25 / 2 =   12  Rest 1
  12 / 2 =    6  Rest 0
   6 / 2 =    3  Rest 0
   3 / 2 =    1  Rest 1
   1 / 2 =    0  Rest 1
```

Das Verfahren endet, wenn das Ergebnis der Division gleich Null ist

Die Binärzahl kann anhand der Reste von unten nach oben ablesen werden: $(1100101010101)_2$

Hinweis: Soll ein solcher Algorithmus implementiert werden, kann der Divisionsrest mittels der Modulo-Operation ermittelt werden: 6485 mod 2 = 1

Umwandlung zwischen Zahlensystemen

Beispiel: Umrechnung der Dezimalzahl 23521 ins Hexadezimalsystem:

```
23521 / 16 =  1470  Rest  1 (1)
 1470 / 16 =    91  Rest 14 (E)
   91 / 16 =     5  Rest 11 (B)
    5 / 16 =     0  Rest  5 (5)
```

Die Ergebnis kann wieder anhand der Reste ablesen werden: $(5BE1)_{16}$

Umwandlung zwischen Zahlensystemen

Beispiel: Umrechnung von $(360)_7$ ins Quinärsystem (Basis 5)

Zunächst muss $(360)_7$ ins Dezimalsystem gewandelt werden:

$(360)_7 = 3 \cdot 49 + 6 \cdot 7 + 0 \cdot 1 = (189)_{10}$

Dann erfolgt die Umrechnung von $(189)_{10}$ ins Quinärsystem:

```
189 / 5 =  37  Rest 4
 37 / 5 =   7  Rest 2
  7 / 5 =   1  Rest 2
  1 / 5 =   0  Rest 1
```

Die Ergebnis kann wieder anhand der Reste ablesen werden: $(1224)_5$

Anstatt den Umweg über das Dezimalsystem zu gehen, hätten wir theoretisch das Verfahren auch direkt im 7er-System anwenden können. Dann müssten wir jedoch die Operationen (Division und Bildung des Rests) im 7-er System ausführen, was wir als Menschen nicht gewohnt sind.

Umwandlung zwischen Binär- und Hexadezimalsystem

Je 4 Bits entsprechen genau einer Hexadezimalziffer

0000 = 0	0100 = 4	1000 = 8	1100 = C
0001 = 1	0101 = 5	1001 = 9	1101 = D
0010 = 2	0110 = 6	1010 = A	1110 = E
0011 = 3	0111 = 7	1011 = B	1111 = F

Damit kann die Umwandlung eine Binärzahl direkt aus den 4-er Blöcken abgelesen werden:

$$(1101\ 0001\ 1110\ 1111)_2 = (D1EF)_{16}$$

Die Hexdezimaldarstellung eignet sich daher besonders, um eine Bitfolge übersichtlich und kompakt darzustellen

Ähnlich leicht ist die Umwandlung einer Binärzahl in andere Systeme mit einer Basis $b = 2^k$, wie z.B. 4 oder 8

"If only dead people understand hexadecimal, how many people understand hexadecimal?"

$$(DEAD)_{16} = 13 \cdot 16^3 + 14 \cdot 16^2 + 10 \cdot 16^1 + 13 \cdot 16^0 = (57005)_{10}$$

Umwandlung bei rationalen Zahlen

Zur Umwandlung von rationalen Zahlen betrachten wir den Vorkommaanteil und Nachkommaanteil getrennt:

$$\begin{aligned} w &= (z_{n-1} \ldots z_2 z_1 z_0, z_{-1} \ldots z_{-m})_b \\ &= \sum_{i=-m}^{n-1} z_i \cdot b^i \\ &= \left(\sum_{i=0}^{n-1} z_i \cdot b^i \right) + \left(\sum_{i=-m}^{-1} z_i \cdot b^i \right) \end{aligned}$$

Die Umrechnung für den Vorkommaanteil ist bekannt, daher betrachten wir nun nur noch den Nachkommaanteil $\tilde{w}$:

$$\tilde{w} = (0, z_{-1} \ldots z_{-m})_b = \left(\sum_{i=-m}^{-1} z_i \cdot b^i \right)$$

Umwandlung bei rationalen Zahlen

Im Gegensatz zum Vorgehen bei der Umwandlung des Vorkommaanteils (Rest der Division) muss beim Nachkommaanteil mit der Basis b multipliziert werden:

$$\tilde{w} = \sum_{i=-m}^{-1} z_i \cdot b^i \Leftrightarrow \tilde{w} \cdot b = b \sum_{i=-m}^{-1} z_i \cdot b^i$$
$$= \left(\sum_{i=-m}^{-2} z_i \cdot b^{i+1} \right) + z_{-1}$$

Der Summand z_{-1} ist der einzige Anteil der ≥ 1 sein kann. Damit ist der Vorkommaanteil von $\tilde{w} \cdot b$ gleich z_{-1}

Der Nachkommaanteil von von $\tilde{w} \cdot b$ stellt den Zahlenwert der verbleibenden Ziffern (ohne z_{-1}) dar. Durch rekursive Anwendung können alle Ziffern bestimmt werden.

Umwandlung bei rationalen Zahlen

Beispiel: Umrechnung der Dezimalzahl 0,6875 ins Binärsystem:

```
0,6875 * 2 = 1,375
0,375  * 2 = 0,75
0,75   * 2 = 1,5
0,5    * 2 = 1,0
```

Das Ergebnis kann aus dem Vorkommaanteil von oben nach unten abgelesen werden: $(0,1011)_2$

Das Verfahren endet, wenn der Nachkommaanteil gleich Null ist

Umwandlung bei rationalen Zahlen

Beispiel: Umrechnung der Dezimalzahl 0,1 ins Binärsystem:

```
0,1 * 2 = 0,2
0,2 * 2 = 0,4
0,4 * 2 = 0,8
0,8 * 2 = 1,6
0,6 * 2 = 1,2
0,2 * 2 = 0,4
0,4 * 2 = ...
```

Der Algorithmus endet nie

Der Nachkommaanteil ist damit im Binärsystem periodisch, obwohl er es im Dezimalsystem nicht ist

Beim Runden auf eine endliche Anzahl von Stellen kommt es daher zu Rundungsfehlern

Umwandlung bei rationalen Zahlen

Rundungsfehler bei der Umwandlung ins Binärsystem müssen bei der Entwicklung von Software und Hardware stets berücksichtigt werden

Ansonsten verhält sich das System anders als erwartet

Beispiel:

```
double a = 0.1;
double b = a * 3;
if(a == 0.1) print("This is ");
if(b == 0.3) print("no"); else print("a");
print(" round-off error\n");
```

Gibt es Fragen?

Anregungen oder Verbesserungsvorschläge können auch gerne per E-mail an mich gesendet werden: Kontakt

Weitere Vorlesungsfolien

Folien auf Englisch (Slides in English)

[Impressum, Datenschutz]