Zahlendarstellung

Benjamin Tröster

Hochschule für Technik und Wirtschaft Berlin

29. November 2021

Fahrplan

Reelle Zahlen

Festkomma-Zahlen

Gleitkommazahlen

Fest- und Gleitkommazahlen

- Zahlendarstellung auf dem Papier:
  - Ziffern: $0, \ldots 9$
  - Vorzeichen: $\pm$
  - Komma: $,$
- Zahlendarstellung im Rechner:
  - Binärziffern: $0, 1$
- $\Rightarrow$ spezielle Vereinbarungen für die Darstellung von Vorzeichen und Komma im Rechner sind erforderlich
- Darstellung des Vorzeichens: wurde im vorigen Abschnitt behandelt
- Darstellung des Kommas: 2 Möglichkeiten
  - Festkommadarstellung (Fixed-point)
  - Gleitkommadarstellung (Floating point)

Festkomma-Zahlen

- Vereinbarung
  - Das Komma sitzt innerhalb des Maschinenwortes, das eine Dualzahl enthalten soll, an einer festen Stelle
  - Meist setzt man das Komma hinter die letzte Stelle.
- Eigenschaften
  - Andere Zahlen können durch entsprechende Maßstabsfaktoren in die gewählte Darstellungsform überführt werden
  - Negative Zahlen: meist Zweierkomplement-Darstellung
  - Festkomma-Darstellungen werden heute hardwareseitig nicht mehr verwendet, jedoch bei Ein- oder Ausgabe!

Festkomma-Zahlen

Datentyp „integer“ (Ganzzahlen) ist ein spezielles Festkommaformat. Manche Programmiersprachen erlauben die Definition von Ganzzahlen unterschiedlicher Länge.

Größe (Bit)	Typische Namen	Vorzeichen	Grenzen des Wertebereichs (Zweierkomplement)	
			min	max
8	char, Byte/byte, modern: **int8_t** bzw. **uint8_t**	signed	−128	127
		unsigned	0	255
16	Word, Short/short, Integer, modern: **int16_t** bzw. **uint16_t**	signed	−32.768	32.767
		unsigned	0	65.535
32	DWord/Double Word, int, long (Windows auf 16/32/64-Bit Systemen; Unix/Linux auf 16/32-Bit Systemen), modern: **int32_t** bzw. **uint32_t**	signed	−2.147.483.648	2.147.483.647
		unsigned	0	4.294.967.295
64	Int64, QWord/Quadword, long long, Long/long (Unix/Linux auf 64-Bit Systemen), modern: **int64_t** bzw. **uint64_t**	signed	−9.223.372.036.854.775.808	9.223.372.036.854.775.807
		unsigned	0	18.446.744.073.709.551.615
128	Int128, Octaword, Double Quadword	signed	$\approx -1{,}70141 \cdot 10^{38}$	$\approx 1{,}70141 \cdot 10^{38}$
		unsigned	0	$\approx 3{,}40282 \cdot 10^{38}$

Gleitkommazahlen

Zur Darstellung von Zahlen, die betragsmäßig sehr groß oder sehr klein sind, verwendet man die Gleitkommadarstellung
Sie entspricht einer halblogarithmischen Form

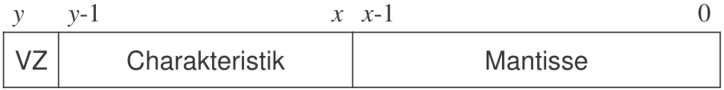
$$X = \pm Mantisse \cdot q^i$$

Die Basis q ist für eine bestimmte Gleitkomma-Darstellung fest (meist 2 oder 16 Bit) und braucht damit nicht mehr explizit repräsentiert zu werden.
Gleitkommazahlen werden meist nicht im Zweierkomplement, sondern mit Betrag und Vorzeichen dargestellt.

Gleitkommazahlen

- Bei der Mantisse ist die Lage des Kommas wieder durch Vereinbarung festgelegt (meist links vom MSB)
- Der Exponent ist eine ganze Zahl, die in Form ihrer Charakteristik dargestellt wird
  - Sowohl für die Charakteristik als auch für die Mantisse wird im Rechner eine feste Anzahl von Speicherstellen festgelegt
- Die Länge der Charakteristik $y - x$ bestimmt die Größe des Zahlenbereichs
- Die Länge der Mantisse x bestimmt die Genauigkeit der Darstellung

Gleitkomma-Maschinenformat



$$\text{Dezimalzahl} = (-1)^{VZ} \cdot (0.\text{Mantisse}) \cdot q^{\text{Exponent}}$$

$$\text{Exponent} = \text{Charakteristik} - q^{(y-1)-x}$$

Normalisierung

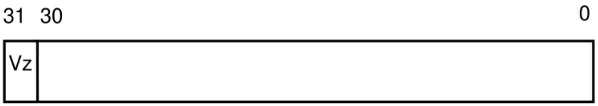
- Eine Gleitkommazahl heißt normalisiert, wenn für den Wert der Mantisse gilt:
$$\frac{1}{q} \leq \mathit{Mantisse} < 1$$
- In dualer Darstellung ist die erste Stelle nach dem Komma gleich 1, d.h. $(0,1\ldots)$
- Ausnahme:
Bei der Zahl 0 sind alle Stellen der Mantisse gleich Null

Normalisierung

- Legt man für die Zahl 0 ein spezielles Bitmuster fest, ist die erste Stelle der Mantisse in normalisierter Form immer gleich 1
- Die erste Stelle der Mantisse braucht im Maschinenformat gar nicht erst dargestellt zu werden, d.h. $(0, 1 \ldots)$
- Man spart ein Bit bei der Speicherung oder gewinnt bei gleichem Speicherbedarf ein Bit an Genauigkeit
- Bei arithmetischen Operationen und bei der Konversion in andere Darstellungen darf diese Stelle natürlich nicht vergessen werden

Beispiel

- Darstellung der Zahl 7135_{10}
- 3 verschiedene Maschinenformate mit je 32 Bit und $q = 2$
- Die Zahl 7135_{10} wird in jedem dieser Formate dargestellt.
  1. Festkommadarstellung mit Zweierkomplement



0 000 0000 0000 0000 0001 1011 1101 1111_2 = 0000 $1BDF_{16}$

Beispiel

2. Gleitkommadarstellung, normalisiert:

31	30 … 23	22 … 0
Vz	Charakteristik	Mantisse

0 100 0110 1 1110 1111 0111 1100 0000 0000_2 = 46EF $7C00_{16}$

3. Gleitkommadarstellung, normalisiert, erste „1“ implizit:

31	30 … 23	22 … 0
Vz	Charakteristik	Rest-Mantisse

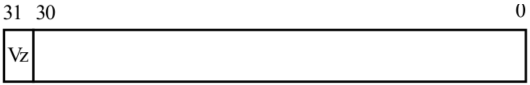
0 1000 1101 101 1110 1111 1000 0000 0000_2 = 46DE $F800_{16}$

Darstellbarer Zahlenbereich

- Die Anzahl darstellbarer Zahlen (Bitkombinationen) ist zwar in allen drei Fällen gleich (2^{32})
- Der Bereich und damit die Dichte darstellbarer Zahlen auf dem Zahlenstrahl ist aber sehr unterschiedlich

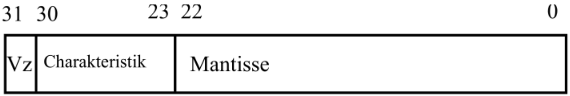
Darstellbarer Zahlenbereich

1.) Zahlen zwischen -2^{31} und $2^{31} - 1$



2.)

- negative Zahlen: $-(1 - 2^{-23}) \cdot 2^{127} \ldots - 0.5 \cdot 2^{-128}$
- positive Zahlen: $0.5 \cdot 2^{-128} \ldots (1 - 2^{-23}) \cdot 2^{127}$
- Null



Darstellbarer Zahlenbereich

3.) normalisierte Gleitkommadarstellung

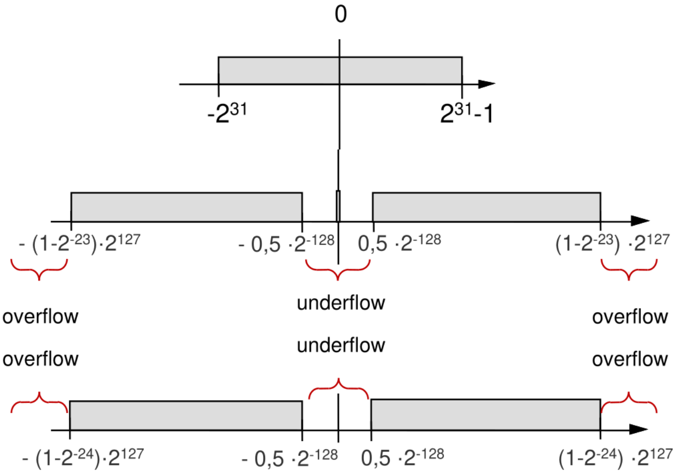
- negative Zahlen: $-(1-2^{-24}) \cdot 2^{127} \ldots -0.5 \cdot 2^{-128}$
- positive Zahlen: $0.5 \cdot 2^{-128} \ldots (1-2^{-24}) \cdot 2^{127}$
- Die Null kann nicht dargestellt werden!

Darstellbarer Zahlenbereich

1.)

2.)

3.)

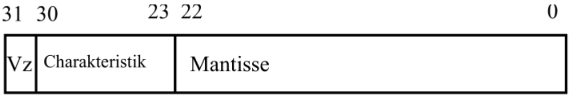


Charakteristische Zahlen

- Um verschiedene Gleitkommadarstellungen miteinander vergleichen zu können, definiert man drei charakteristische Zahlen:
  - `maxreal` ist die größte darstellbare normalisierte positive Zahl
  - `minreal` ist die kleinste darstellbare normalisierte positive Zahl
  - `smallreal` ist die kleinste Zahl, die man zu 1 addieren kann, um einen von 1 verschiedenen Wert zu erhalten

Charakteristische Zahlen – Beispiel

- In Format 2.) im letzten Beispiel

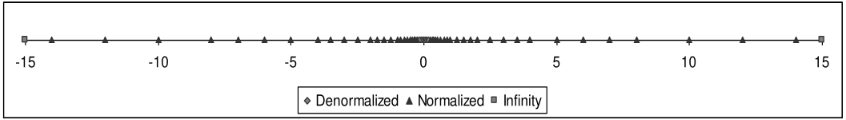


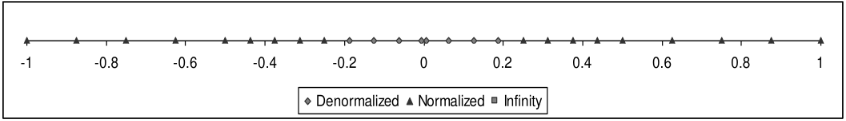
  - maxreal $= (1 - 2-^{23}) \cdot 2^{127}$
  - minreal $= 0.5 \cdot 2^{-128}$
- Die Zahl 1 wird normalisiert als $0.5 \cdot 2^1$ dargestellt
- Die nächstgrößere darstellbare Zahl hat in der Mantisse zusätzlich zur 1 in Bit 22 eine 1 in Bit 0
- smallreal $= 0.00000000000000000000001_2 \cdot 2^1$,
- also smallreal $= 2^{-23} \cdot 2^1 = 2^{-22}$

Ungenauigkeiten

- Die Differenz zwischen zwei aufeinanderfolgenden Zahlen wächst bei Gleitkomma-Zahlen exponentiell mit der Größe der Zahlen, während sie bei Festkommazahlen konstant ist.
- Bei der Darstellung großer Zahlen ergibt sich damit auch eine hohe Ungenauigkeit
- Die Gesetzmäßigkeiten, die für reelle Zahlen gelten, werden für Maschinendarstellungen verletzt!
- (auch wenn diese Zahlen in einer höheren Programmiersprache oft `real` heißen).

Ungenauigkeiten





Beispiel Ungenauigkeit

- Das Assoziativgesetz $(x + y) + z = x + (y + z)$ gilt selbst dann nicht allgemeingültig, wenn kein overflow oder underflow auftritt
  - z.B.: $x = 1; y = z = \mathit{smallreal}/2$

$$
\begin{aligned}
(x + y) + z &= (1 + \mathit{smallreal}/2) + \mathit{smallreal}/2 \\
&= 1 + \mathit{smallreal}/2 \\
&= 1 \\
x + (y + z) &= 1 + (\mathit{smallreal}/2 + \mathit{smallreal}/2) \\
&= 1 + \mathit{smallreal} \\
&\neq 1
\end{aligned}
$$

Quellen I

Hoffmann, Dirk W (2020). *Grundlagen der technischen Informatik*. Carl Hanser Verlag GmbH Co KG.