1. Informationssysteme

In der Informatik unterscheidet man zwischen der Information und der Nachricht durch die sie übermittelt wird. Formalisiert wird diese Unterscheidung durch den Begriff des Informationssystems:

Ein Informationssystem ist ein Tripel $(\mathsf{R}, \mathsf{A}, \mathrm{I})$ bestehend aus

- einer Menge R von *Repräsentationen* (Informationsdarstellungen, Nachrichten),
- einer Menge A von *Bedeutungen* (Informationen, Abstraktionen) und
- einer *Interpretationsfunktion* $\mathrm{I} : \mathsf{R} \to \mathsf{A}$ (Abstraktionsfunktion).

Repräsentationen sind immer konkrete Phänomene (Zeichen auf einem Blatt Papier, elektromagnetische Wellen, etc.), Informationen sind abstrakte Ideen. Die Interpretation ist die Verbindung zwischen den beiden: Informationen können nur mittels einer Repräsentation gespeichert, kommuniziert oder verarbeitet werden; Repräsentation ohne zugeordnete Information ist sinn-los.

Da Informationen abstrakte Ideen sind, lässt sich die Interpretationsfunktion nicht aufschreiben! Man kann nur eine Abbildung von einer Repräsentation R_1 in eine andere Repräsentation R_2 angeben. Wenn R_2 „einfacher“ als R_1 ist lässt sich dadurch die Interpretationsfunktion annähern.

Wie erlernt man die Bedeutung neuer Symbole und Zeichen? Die Bedeutung neuer, noch unbekannter, Repräsentationen erlernt man häufig durch Übersetzung in bereits bekannte Repräsentationen, von denen man annimmt, dass die Bedeutung klar ist, z.B., mathematische Formalismen oder natürliche Sprache. Wenn man versucht die Bedeutung durch natürliche Sprache zu definieren kann es zu Missverständnissen kommen.

Eine Programmiersprache kann man z.B. durch

- eine formale mathematische Semantik,
- natürlichsprachliche Beschreibungen oder
- Ausprobieren am Computer

erlernen.

Aber: Die eigentliche Information ist nur in unserem Kopf. (Man kann nie vor Missverständnissen sicher sein.)

2. Hinweise zur Notation

Wenn man Zahlen, Texte, Graphiken oder irgendwelche anderen Daten auf einem Computer verarbeiten will, muss man sie in einer geeigneten Repräsentation darstellen. Dabei gibt es viele denkbare Möglichkeiten eine Repräsentationen zu wählen, z.B. graphisch, als Formel (Abbildung 1) oder als Sprachaufzeichnung.

$$\oint \frac{\left(2x_{i_j}\right)^3 - 7y^{k_{i_j}} + f(x,y)}{b_{i_j} + \nabla^2 g(x)} \partial x \partial y = \int_{-\infty}^{\infty} \nu(x) dx$$

Abbildung 1. Mathematische Formel

Für viele Daten ist es sinnvoll sie durch lineare Zeichenketten darzustellen, z.B. Zahlen oder kurze Textstücke.

```
123
To be or not to be
ASDFGHJKL
```

Eine derartige lineare Darstellung ist nicht immer die „beste“ Möglichkeit die Information zu repräsentieren, so ist für die meisten Leser die Schreibweise

$$\frac{x^2+2yz}{3a+b} = 17v_i$$

wesentlich leichter zu verstehen als die (vom Informationsgehalt her gleichwertige) lineare Form

```
\frac{x^2+2yz}{3a+b} = 17v_i
```

Für den Computer sind jedoch Notationen in denen die räumlichen Beziehungen zwischen den einzelnen Zeichen eine Rolle spielt schwerer zu verarbeiten. Daher verwendet man in der Informatik oft lineare Anordnungen von Symbolen als Repräsentation von Informationen. Man kann z.B. auch einen mehrzeiligen Text in eine lineare Form überführen, indem man ein eigenes Zeichen für den Zeilenwechsel einführt. Schreibt man z.B $\oplus$ für einen Zeilenwechsel, so kann man den Text

> Toi qui, comme un coup de couteau
> Dans mon cœur plaintif es entrée

auch repräsentieren als

> Toi qui, comme un coup de couteau⊕Dans mon cœur plaintif es entrée

2.1. **Lineare Repräsentationen.** Da lineare Zeichenketten in der Informatik sehr häufig vorkommen führt man dafür spezielle Bezeichnungen bzw. Notationen ein:

Die Menge der einzelnen Zeichen, die in den Wörtern vorkommen können, nennt man das zugrundeliegende *Alphabet*. Man sagt z.B. die natürlichen Zahlen (in der üblichen Schreibweise) sind Worte über dem Alphabet $\mathcal{Z} = \{0,1,2,3,4,5,6,7,8,9\}$

Hat man ein Alphabet $\mathcal{A}$ gegeben, so schreibt man $\mathcal{A}^*$ für die Menge aller Worte die durch Aneinanderreihung von Zeichen aus $\mathcal{A}$ gebildet werden können, man sagt $\mathcal{A}^*$ ist die Menge aller Worte über $\mathcal{A}$. Z.B. ist $\mathcal{Z}^*$ die Menge aller Zahlendarstellungen in der üblichen Notation, mit führenden Nullen: $1435 \in \mathcal{Z}^*$, $000012 \in \mathcal{Z}^*$. Ist $\mathcal{B}$ das Alphabet $\{0,1\}$, so ist $\mathcal{B}^*$ die Menge aller Worte die nur aus 0 und 1 bestehen. Es sind z.B. 010101, 1111, 000000 Worte aus $\mathcal{B}^*$.

Es ist sinnvoll einen Spezialfall in die Definition von $\mathcal{A}^*$ mit aufzunehmen: Die Definition von A^* im letzten Absatz legt die Annahme nahe, dass jedes Wort aus A^* mindestens ein Zeichen enthalten muss. Es ist aber zweckmäßiger festzulegen, dass auch das *leere Wort*, also ein Wort das gar kein Zeichen enthält in $\mathcal{A}^*$ entalten ist. Man bezeichnet dieses Wort oft durch den griechischen Buchstaben ε (Epsilon).

Wenn man irgendwelche Aussagen über ein Wort treffen will, so muss man sich dazu oft auf die einzelnen Zeichen, aus denen es sich zusammensetzt, beziehen. Bei jedem konkret gegebenen Wort ist das leicht möglich: Spricht man über die Zahl 3435 so kann man von der Ziffer 4 oder dem ersten Vorkommen der Ziffer 3 reden. Will man aber eine Aussage über alle Worte einer Sprache oder über ein nicht näher bestimmtes Wort machen, so benötigt man eine Notation mit der das leicht möglich ist. Ein häufig verwendetes Hilfsmittel ist, dem Wort einen Namen zu geben. Man sagt z.B. „Sei w ein Wort aus $\mathcal{A}^*$.“ oder „Für ein beliebiges $w \in \mathcal{A}^*$ gilt ...“

Wir wissen, dass (per Definition) jedes Wort $w \in \mathcal{A}^*$ aus Zeichen besteht, die in $\mathcal{A}$ vorkommen. Wir wissen aber nicht, wie viele Zeichen in w vorkommen; w kann aus einem einzelnen Zeichen bestehen, w kann aus vielen Millionen Zeichen bestehen

oder w kann sogar das leere Wort sein und aus gar keinem Zeichen bestehen. Man schreibt daher oft „Sei $w = w_1 \dots w_n \in \mathcal{A}^*$“ oder „Für jedes $w = w_1 \dots w_n \in \mathcal{A}^*$ gilt ...“. Diese Notation bedeutet folgendes: w ist ein Wort aus $\mathcal{A}^*$, d.h. w ist eine (endliche) Folge von Symbolen aus $\mathcal{A}$. Ist w nicht das leere Wort, (in Zeichen: $w \neq \varepsilon$), so besteht w aus einer gewissen Anzahl von Zeichen, wir nennen diese Zahl n. Dann können wir dem Zeichen, das in w an der ersten Stelle steht den Namen w_1 zuordnen, falls $n > 1$ ist können wir dem Zeichen an der zweiten Stelle den Namen w_2 zuordnen, usw. Um auch das leere Wort mit dieser Notation beschreiben zu können setzt man fest, dass man mit $w_1 \dots w_n$ das leere Wort meint falls $n = 0$.

Die Wahl der Buchstaben spielt dabei natürlich keine Rolle, man kann auch schreiben: „Sei $w = a_1 \dots a_k \in \mathcal{A}^*$ oder „Seien $v = t_1 \dots t_k$ und $w = s_l \dots s_m$ Worte über $\mathcal{A}$.“ Allerdings muss man im letzten Fall verschiedene Indizes (k und m) verwenden wenn man für v und w unterschiedliche Längen zulassen will. Schreibt man „Seien $v = v_1 \dots v_n$ und $w = w_1 \dots w_n$ Worte über $\mathcal{A}$,“ so sind v und w Worte mit unbestimmter *aber gleicher* Länge, d.h., wir wissen nicht aus wie vielen Zeichen v besteht, aber w besteht aus genau so vielen Zeichen wie v.

Es ist nicht erforderlich das erste Zeichen mit dem Index 1 zu benennen. Man kann ebenso die Zählung mit 0 oder mit 42 anfangen. Wenn man 0 als ersten Index nimmt, schreibt man häufig als letzten Index nicht n sondern $n-1$ also z.B. „Sei $w_0 \dots w_{n-1} \in \mathcal{A}^*$.“ Diese Konvention hat den Vorteil, dass die Länge des Wortes immer noch n ist. Man verallgemeinert die Notation für das leere Wort, indem man festlegt, dass $w_i \dots w_k$ das leere Wort darstellt, sobald $k < i$ ist.

2.2. **Operationen mit Indizes.** Wenn man ein Wort w in der Form $w_1 \dots w_n$ darstellt kann man viele Operationen in kompakter Schreibweise darstellen. Will man z.B. das Wort v durch Streichen des ersten Buchstabens aus w erhalten, so kann man schreiben $v = w_2 \dots w_n$. Falls $n = 1$ ist, so ist v das leere Wort. Gemäß der im letzten Abschnitt genannten Konvention ist aber $w_2 \dots w_1$ nur eine komplizierte Schreibweise für das leere Wort, so dass wir auch in diesem Fall das korrekte Ergebnis erhalten.

Ebenso kann man $v = w_1 \dots w_{n-1}$ für das Wort schreiben, das man durch aus w erhält, indem man den letzten Buchstaben streicht. Streicht man das an der Stelle i vorkommende Zeichen aus w, so kann man das dadurch entstehende Wort als $w_1 \dots w_{i-1} w_{i+1} \dots w_n$ schreiben. Um diese Operation auf das Streichen beliebig vieler Zeichen aus w zu verallgemeinern benötigt man den Begriff der aufsteigenden Folge.

Eine aufsteigende Folge natürlicher Zahlen ist eine Folge natürlicher Zahlen, in der jede Zahl (echt) größer ist, als ihr Vorgänger. Z.B ist $1, 534, 234, 1000, 10001$ eine aufsteigende Folge natürlicher Zahlen, nicht jedoch $3, 7, 4, 9$ und auch nicht $5, 7, 7, 10$.

Mit aufsteigenden Folgen kann man Worte beschreiben, die aus einem gegebenen Wort w durch Streichen einzelner Zeichen entstehen. Seien $w = w_1 \dots w_n$ ein Wort über einem Alphabet $\mathcal{A}$ und $J = j_1, \dots j_k$ eine aufsteigende Folge von Zahlen aus der Menge $\{1, \dots, n\}$. Dann kann man ein neues Wort v bilden, indem man diejenigen w_i, deren Index i in J vorkommt, in der Reihenfolge ihres Vorkommens in w aneinandersetzt. Anders ausgedrückt: v entsteht aus w indem man alle Zeichen w_i, deren Index i *nicht* in J vorkommt aus w streicht.

Man kann das so gebildete Wort als $v = w_{j_1} \dots w_{j_k}$ schreiben. Ist z.B. $w = w_1 \dots w_5 = \mathsf{SAMBA}$ und $J = j_1, j_2, j_3 = 1, 3, 4$ so hat man $w_{j_1} w_{j_2} w_{j_3} = w_1 w_3 w_4 = \mathsf{SMB}$.

3. Zahlensysteme

3.1. **Dualzahlen (Binärzahlen).** Wie viele Zahlen können mit Bitfolgen der Länge 3 dargestellt werden?

Antwort: So viele Zahlen, wie es (verschiedene) 3-er-Kombinationen aus 0 und 1 gibt, das sind $2^3 = 8$ Zahlen.

$$\begin{array}{ll} 000_2 & = 0_{10} \\ 001_2 & = 1_{10} \\ 010_2 & = 2_{10} \\ 011_2 & = 3_{10} \end{array} \qquad \begin{array}{ll} 100_2 & = 4_{10} \\ 101_2 & = 5_{10} \\ 110_2 & = 6_{10} \\ 111_2 & = 7_{10} \end{array}$$

Allgemein können mit Bitfolgen der Länge n genau 2^n Zahlen, 0, 1, $\ldots$, $2^n - 1$ dargestellt werden. Der Binärzahl

$$Z_{n-1} \ldots Z_1 Z_0$$

entspricht die ganze Zahl

$$2^{n-1} Z_{n-1} + \cdots + 2Z_1 + Z_0.$$

Aufgabe 1. Der Binärzahl 1011_2 entspricht die Dezimalzahl

$$\begin{array}{lccccccc} & 1 \cdot 2^3 & + & 0 \cdot 2^2 & + & 1 \cdot 2^1 & + & 1 \cdot 2^0 \\ = & 8 & + & 0 & + & 2 & + & 1 \\ = & 11 & & & & & & \end{array}$$

Der Binärzahl 11010_2 entspricht

$$\begin{array}{lccccccccc} & 1 \cdot 2^4 & + & 1 \cdot 2^3 & + & 0 \cdot 2^2 & + & 1 \cdot 2^1 & + & 0 \cdot 2^0 \\ = & 16 & + & 8 & & & + & 2 & & \\ = & 26 & & & & & & & & \end{array}$$

Der Binärzahl 101101_2 entspricht

$$\begin{array}{lccccccc} & 1 \cdot 2^5 & + & 1 \cdot 2^3 & + & 1 \cdot 2^2 & + & 1 \cdot 2^0 \\ = & 32 & + & 8 & + & 4 & + & 1 \\ = & 45 & & & & & & \end{array}$$

3.2. **Spiel: Zahlen erraten.** Beim Spiel „Zahlen erraten“ hat man n Karten, auf denen Zahlen zwischen 1 und $2^n - 1$ stehen. Der Spieler bittet einen Mitspieler sich eine dieser Zahlen zu denken und dem Spieler mitzuteilen, auf welchen Karten die Zahl steht. Aus diesen Karten kann der Spieler die gedachte Zahl berechnen. Wie müssen die Zahlen auf den Karten angeordnet sein, damit das möglich ist?

Eine mögliche Lösung ist die folgende: Eine Zahl z mit $0 \leq z \leq 2^n - 1$ befindet sich genau dann auf der i-ten Karte ($i = 0, \ldots, n-1$), wenn in der Binärcodierung $b_{n-1} \ldots b_0$ von z gilt: $b_i = 1$.

Zur Decodierung muss man 2^i für die i-te Karte wissen. In der Binärdarstellung von 2^i ist das Bit an der Stelle i gleich 1, alle anderen Bits sind gleich 0. Bei allen anderen Zahlen auf der Karte ist das Bit an der Stelle i gleich 1 und mindestens ein zusätzliche Bits ist von 0 verschieden. Daher ist 2^i die kleinste Zahl auf der i-ten Karte.

3.3. **Hexadezimalzahlen.** Mit 2 Zeichen aus

$$H = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F$$

können $16^2 = 256$ Zahlen dargestellt werden, also Ebensoviele wie mit einem Byte (8 Bit) Mit 3 Zeichen aus H können $16^3 = 4096$ Zahlen dargestellt werden.

Der Hexadezimalzahl $Z_{n-1} \dots Z_1 Z_0$ entspricht die ganze Zahl

$$Z_{n-1} \cdot 16^{n-1} + \cdots + Z_1 \cdot 16 + Z_0$$

wobei man die Buchstaben A bis F als zusätzliche Ziffern verwendet:

$$\begin{array}{lcllcl} A_{16} & = & 10_{10} & D_{16} & = & 13_{10} \\ B_{16} & = & 11_{10} & E_{16} & = & 14_{10} \\ C_{16} & = & 12_{10} & F_{16} & = & 15_{10} \end{array}$$