

---

# 1. Wörter und Sprachen

---

Als Daten werden wir Wörter über einem festen Alphabet betrachten. Probleme werden dann durch Sprachen, d.h. Mengen von Wörtern repräsentiert. In diesem Abschnitt führen wir diese Konzepte ein und fassen deren später benötigte Eigenschaften zusammen.

**1.1 DEFINITION.** Ein *Alphabet*  $(\Sigma, <)$  ist eine endliche, nichtleere Menge  $\Sigma$  zusammen mit einer totalen Ordnung  $<$  auf  $\Sigma$ . Die Elemente von  $\Sigma$  heißen die *Buchstaben* (*Symbole*) des Alphabets.

Ein *Wort* über dem Alphabet  $(\Sigma, <)$  ist eine Folge über  $\Sigma$ , d.h. eine endliche Folge von Buchstaben aus  $\Sigma$ . Die Menge aller Wörter über  $\Sigma$  wird mit  $\Sigma^*$  bezeichnet.

Eine *Sprache* über dem Alphabet  $(\Sigma, <)$  ist eine Menge von Wörtern über diesem Alphabet, d.h. eine Teilmenge von  $\Sigma^*$ .

Kleine Buchstaben  $a, b, c$  vom Anfang des Alphabets reservieren wir für Buchstaben, kleine Buchstaben  $u, v, w, x, y, z$  vom Ende des Alphabets für Wörter. Sprachen bezeichnen wir mit großen (kursiven) Buchstaben  $A, B, C, \dots$ , Mengen von Sprachen, die wir auch kurz *Klassen* nennen, mit großen (normalen) Buchstaben  $A, B, C, \dots$ . Kleine Buchstaben  $k, l, m, n$  aus der Mitte des Alphabets werden in der Regel natürliche Zahlen beschreiben.

Geben wir ein Alphabet  $(\Sigma, <)$  durch Auflisten der Buchstaben an, so geschieht dies immer bzgl. der zugrundegelegten Ordnung: Für  $\Sigma = \{a_0, \dots, a_n\}$  gilt  $a_0 < a_1 < \dots < a_n$ . Häufig lassen wir in der Bezeichnung eines Alphabetes die Ordnung  $<$  weg, falls diese implizit bekannt oder (an dieser Stelle) unwesentlich ist.

Ein Alphabet mit  $n$  Buchstaben heißt *n-äres Alphabet* und wird mit  $\Sigma_n$  bezeichnet. Insbesondere sind  $\Sigma_1 = \{0\}$  und  $\Sigma_2 = \{0, 1\}$  das *unäre* bzw. *binäre* Alphabet.

Die leere Folge über einem Alphabet  $\Sigma$  nennen wir das *leere Wort* und schreiben hierfür  $\lambda$ . Die *Länge*  $|w|$  eines Wortes  $w$  ist die Anzahl der in  $w$  vorkommenden Buchstaben, d.h.  $|\lambda| = 0$  und  $|b_1 \dots b_n| = n$  für  $w = b_1 \dots b_n$  mit  $b_k \in \Sigma$  ( $1 \leq k \leq n$ ). Ist  $w$  ein Wort der Länge  $n$ , so bezeichnen wir mit  $w(k)$  den  $k+1$ -ten Buchstaben von  $w$  ( $k < n$ ), d.h.  $w = w(0) \dots w(1)$ .

Weiter benutzen wir folgende Notation

- $\Sigma^+ = \Sigma^* - \{\lambda\}$  ist die Menge der nichtleeren Wörter über dem Alphabet  $\Sigma$ .
- $\Sigma^n = \Sigma^{=n} = \{w \in \Sigma : |w| = n\}$  ist die Menge der Wörter der Länge  $n$  über  $\Sigma$ .

Entsprechend enthalten  $\Sigma^{<n}$  und  $\Sigma^{\leq n}$  die Wörter über dem Alphabet  $\Sigma$ , deren Länge kleiner bzw. kleiner oder gleich  $n$  ist. Man beachte, dass es  $k^n$  Wörter der Länge  $n$  über dem  $k$ -ären Alphabet  $\Sigma_k$  gibt, also

$$|\Sigma_k^n| = k^n \tag{1.1}$$



$w$  liegt, falls  $v$  auf dem Weg von der Wurzel  $\lambda$  zu dem Knoten  $w$  liegt, so liegt  $v$  genau dann oberhalb von  $w$ , wenn  $v$  Anfangsstück von  $w$  ist.

Dass der Knoten  $v$  links vom Knoten  $w$  liegt, lässt sich durch

$$v <_L w \quad :\Leftrightarrow \quad \exists x, y, z \in \Sigma^* \exists a, b \in \Sigma (a < b \ \& \ v = xay \ \& \ w = xbz)$$

beschreiben. Die Knoten, die links oder echt oberhalb von einem Knoten  $w$  liegen, sind genau die mit den lexikographisch kleineren Wörtern markierten Knoten:

$$v <_{\text{lex}} w \quad :\Leftrightarrow \quad v <_L w \vee v \sqsubset w$$

Bei der *kanonischen* (oder *längenlexikographischen*) *Ordnung* auf  $\Sigma^*$  wählt man die Wortlänge als Haupt- und die lexikographische Ordnung als Nebenordnungskriterium:

$$v < w \quad :\Leftrightarrow \quad |v| < |w| \vee (|v| = |w| \ \& \ v <_{\text{lex}} w)$$

Für jedes Alphabet  $\Sigma$  erhält man hierdurch eine lineare Ordnung auf  $\Sigma^*$ , die isomorph zur Ordnung der natürlichen Zahlen ist, d.h. eine Nummerierung der Wörter erlaubt. Für  $k = 1$  bzw.  $k = 2$  beginnt diese Ordnung mit

$$\lambda < 0 < 00 < 000 < \dots$$

bzw.

$$\lambda < 0 < 1 < 00 < 01 < 10 < 11 < 000 < \dots$$

Mit  $w_n^\Sigma$  bezeichnen wir das  $(n+1)$ -te Wort von  $\Sigma^*$  bzgl. der kanonischen Ordnung. Statt  $w_n^{\Sigma_1}$  und  $w_n^{\Sigma_2}$  schreiben wir meist  $\text{Un}(n)$  bzw.  $\text{Bin}(n) = \underline{n}$ . Man beachte, dass wegen (1.1) die Länge von  $\text{Un}(n)$  linear in  $n$  ist, wogegen für mehrbuchstabiges Alphabet  $\Sigma$  die Länge von  $w_n^\Sigma$  logarithmisch in  $n$  ist. Für  $\text{Un}(n)$  und  $\underline{n} = \text{Bin}(n)$  liefern (1.2) und (1.3) die Größen

$$|\text{Un}(n)| = n \tag{1.4}$$

$$2^{|\text{Bin}(n)|} - 1 \leq n \leq 2^{|\text{Bin}(n)|+1} - 1 \tag{1.5}$$

Zum Vergleich des Wachstumsverhaltens von Funktionen werden wir folgende Begriffe über asymptotisches Verhalten und Größenordnung einer Funktion verwenden. Für Funktionen  $f$  und  $g$  von  $\mathbb{N}$  nach  $\mathbb{N}$  definieren wir

$$f \leq g \quad :\Leftrightarrow \quad \forall n \in \mathbb{N} (f(n) \leq g(n))$$

$$f \leq_{\text{f.ü.}} g \quad :\Leftrightarrow \quad \begin{aligned} &\forall n \in \mathbb{N} (f(n) \leq g(n)), \text{ d.h.} \\ &\exists n_0 \in \mathbb{N} \forall n \geq n_0 (f(n) \leq g(n)) \end{aligned}$$

$$f \in O(g) \quad :\Leftrightarrow \quad \begin{aligned} &\exists c, d \in \mathbb{N} \forall n \in \mathbb{N} (f(n) \leq cg(n) + d) \\ &\Leftrightarrow \quad \exists c, d \in \mathbb{N} (f \leq cg + d) \end{aligned}$$

$$f \in o(g) \quad :\Leftrightarrow \quad \begin{aligned} &\forall c \in \mathbb{N}_+ \forall n \in \mathbb{N} (f(n) \leq \frac{1}{c}g(n)) \\ &\Leftrightarrow \quad \forall c \in \mathbb{N}_+ (f \leq_{\text{f.ü.}} \frac{1}{c}g) \end{aligned}$$

und wir sagen entsprechend "f ist kleiner-gleich g", "f ist fast überall oder asymptotisch kleiner-gleich g", "f ist höchstens von der Ordnung g", "f ist von (echt) kleinerer

Ordnung als  $g$ ". (Hierbei bezeichnet  $\mathbb{N}$  die Menge der natürlichen Zahlen einschließlich der Null und  $\mathbb{N}_+ = \mathbb{N} - \{0\}$ .)

Die Relationen  $<$  und  $<_{f.ü.}$  etc. auf Funktionen sind entsprechend definiert. Gilt  $O(f) = O(g)$ , so sagen wir, dass  $f$  und  $g$  von der gleichen Ordnung sind. Für  $g \geq_{f.ü.} 1$  zeigt man leicht

$$\begin{aligned} f \in O(g) &\Leftrightarrow \exists c \in \mathbb{N} (f \leq_{f.ü.} cg) \\ &\Leftrightarrow \limsup_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty \\ f \in o(g) &\Leftrightarrow \lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \end{aligned}$$

Bei unseren Untersuchungen wird es häufig genügen, die Größenordnung einer Funktion zu bestimmen. Kodieren wir Daten eines anderen Typs durch Wörter, so wird zum Beispiel die Größenordnung der Wortlängen der kodierten Daten wichtig sein. Repräsentieren wir z.B. eine natürliche Zahl  $n$  durch das Unärwort  $Un(n)$  oder das Binärwort  $Bin(n) = \underline{n}$ , so haben die Längen dieser Darstellungen nach (1.4) und (1.5) unterschiedliche Größenordnungen. Benutzen wir für die Kodierung ein mehrbuchstabiges Alphabet, so lässt sich die Größenordnung im Vergleich zum binären Alphabet jedoch nicht mehr verbessern. Dies ergibt sich unmittelbar aus folgender Binärcodierung von Wörtern über mehrbuchstabigem Alphabet  $\Sigma_k$  ( $k > 2$ ).

1.2 DEFINITION. Sei  $\Sigma_k = \{a_1, \dots, a_k\}$  ein Alphabet mit  $k > 2$  Buchstaben. Die durch

$$\text{bin}_k(a_{i_1} \dots a_{i_m}) = 0^{i_1} 1^{k-i_1} 0^{i_2} 1^{k-i_2} \dots 0^{i_m} 1^{k-i_m}$$

definierte Abbildung  $\text{bin}_k : \Sigma_k^* \rightarrow \Sigma_2^*$  heisst *Binärcodierung* der Wörter über dem Alphabet  $\Sigma_k$ .

Offensichtlich ist  $\text{bin}_k$  injektiv und es gilt  $|\text{bin}_k(w)| = k|w|$ , d.h.  $|\text{bin}_k(w)| \in O(|w|)$ . Weiter lassen sich  $\text{bin}_k$  und die Umkehrfunktion  $\text{bin}_k^{-1}$  sehr leicht berechnen (s. Übungen). Dies rechtfertigt, dass man Daten i.a. binär kodiert. Insbesondere werden im folgenden zahlentheoretische Probleme  $P \subseteq \mathbb{N}$  mit ihrer Binärdarstellung  $P_{\text{bin}} = \{\underline{n} : n \in P\} \subseteq \Sigma_2^*$  identifiziert. Dies ist wesentlich, wenn wir die Komplexität dieser Probleme bestimmen, da wir die Kosten eines Verfahrens als Funktion in der Eingabelänge ausdrücken werden.