2 Lauter Wörter

Informatik ist ohne Zeichenketten, die in der Literatur oft auch Wörter genannt werden, undenkbar. Auch die Theorie ist stark auf sie angewiesen. Im vorigen Abschnitt spielen sie bereits als mögliche Abläufe, *an-aus*-Zyklen und verbotene Ereignisfolgen eine wichtige Rolle. Man muss sie hintereinander schreiben und Teilfolgen identifizieren können; man muss Gleichheit von Ereignisfolgen feststellen können. Den Benutzerinnen und Benutzern von Programmiersprachen sind Zeichenketten als Namen und Ausdrücke, als Dateien und Felder (Ketten konstanter Länge) u.ä. sowie in Gestalt der Programme selbst geläufig. Wörter und Sätze einer natürlichen Sprache wie Deutsch oder Englisch sind weitere wichtige Beispiele.
In diesem Abschnitt sind einige wichtige Informationen zum Umgang mit Zeichenketten zusammengestellt. Vieles davon wird verschiedentlich in die Überlegungen zur theoretischen Informatik während des kommenden Semesters eingehen, ohne dass diese Tatsache immer ausführlich gewürdigt wird.

2.1 Erzeugung von Wörtern

1. Um nicht bei jedem einzelnen Wort den Rahmen festlegen zu müssen, wird vorweg ein Zeichenvorrat A ausgewählt. A wird auch *Alphabet*, die Elemente von A werden *Zeichen* oder *Symbole* genannt.

2. *Wörter* (über A) sind rekursiv definiert durch:

   (a) λ ist ein *Wort*,

   (b) mit $x \in A$ und einem Wort v ist auch xv ein *Wort*.

3. Das initiierende Wort in (a) wird *leeres Wort*, der wiederholbare Vorgang in (b) *Linksaddition* (von x zu v) genannt. Für Wörter sind auch die Bezeichnungen Zeichenketten, Listen, Folgen, Sequenzen, Sätze, "files", Texte, Nachrichten, "strings" u.v.a.m. gebräuchlich. Die erste Bezeichnung wird im folgenden synonym für Wörter verwendet.

   Die Menge aller Wörter über A wird mit A^* bezeichnet. Für die Linksaddition $x\lambda$ von x zu λ wird kurz auch x geschrieben. In diesem Sinne gilt: $A \subseteq A^*$.

   Der Erzeugungsprozess für Wörter ist in Abbildung 1 illustriert.

4. Beispielsweise entsteht für das Alphabet $\{0,1\}$ anfangs nur das leere Wort λ, weil der Teil (b) des Worterzeugungsprozesses nur verwendet werden kann, wenn schon Wörter vorhanden sind. Der Teil (a) muss nicht wieder angewendet werden, weil dadurch keine neuen Wörter mehr entstehen können. Die Anwendung von Teil (b) liefert nun zwei neue Wörter: $0\lambda, 1\lambda$ (bzw. $0, 1$ nach der Konvention zur Linksaddition mit dem leeren Wort). Darauf kann der Teil (b) erneut angewendet werden, was vier neue Wörter liefert: 00, 01, 10, 11. Durch Fortsetzung des Verfahrens (ad infinitum) entstehen nach und nach alle (endlichen) Bitstrings.

5. Die Erzeugung von Wörtern ist so gemeint, dass nur Gebilde, die durch den in Punkt 2 gegebenen rekursiven Prozess entstehen, Wörter über A sind und dass jedes Wort in eindeutiger Weise aus diesem Prozess hervorgeht. Letzteres bedeutet, dass für jedes

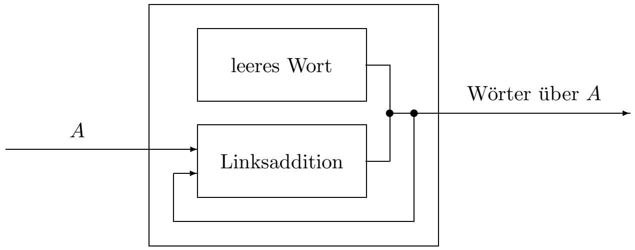

Abbildung 1: Rekursive Erzeugung von Wörtern

Wort w entweder $w = \lambda$ gilt oder eindeutig $x \in A$ und ein Wort v mit $w = xv$ existieren. Im zweiten Fall wird x mit $head(w)$ und v mit $tail(w)$ bezeichnet.

Bei dem Erzeugungsprozess wird stillschweigend vorausgesetzt, dass man durch die Bezeichnung λ ein Gebilde erhält, das sich von allen anderen Wörtern unterscheiden lässt, und dass das Nebeneinanderschreiben von Zeichen und Wörtern möglich ist. Beides mag intuitiv klar sein; es wird hier einfach vorausgesetzt.

Wenn das Alphabet selbst wieder Wörter enthält, wie z.B. $\{E, I, EI\}$, muss man mit dem Nebeneinanderschreiben aufpassen. Denn sonst kann beispielsweise EI ein Wort aus einem wie zwei Zeichen sein, was wegen der verlangten Eindeutigkeit verboten ist. In solchen Fällen muss man extra Vorsorge treffen, indem man die Zeichen in Hochkommata oder sonstige Klammern einschließt oder Zeichen und Wort beim Nebeneinanderschreiben durch ein Blank oder ein sonstiges Trennzeichen auseinanderhält.

Das durch diese Vereinbarungen verfügbare Textsystem ist noch recht bescheiden. Beginnend mit dem leeren Wort, kann jedes Wort von rechts nach links geschrieben werden; das zuletzt geschriebene Symbol kann gelesen (*head*) oder gelöscht (*tail*) werden. Wünschenswert wäre mehr Komfort, was mit den Punkten 2.2 und 2.4 ein Stück weit erreicht wird und sich analog noch wesentlich weitertreiben ließe.

2.2 Konkatenation

1. Analog zur Linksaddition und allgemeiner als diese lassen sich auch zwei Wörter v, w zu einem neuen Wort $v \cdot w$ zusammensetzen. $v \cdot w$ wird *Konkatenation* von v und w genannt und ist folgendermaßen rekursiv definiert:

   (a) für $v = \lambda$ gilt $\lambda \cdot w = w$,

   (b) für $v = xu$ mit $x \in A$ gilt $(xu) \cdot w = x(u \cdot w)$.

2. Die Linksaddition erweist sich als Spezialfall der Konkatenation:

$$xv \underset{1a}{=} x(\lambda \cdot v) \underset{1b}{=} (x\lambda) \cdot v \underset{2.1.3}{=} x \cdot v.$$

   Das rechtfertigt die Schreibweise vw für $v \cdot w$.

3. Die Konkatenation fügt zwei Wörter zusammen. Meist werden jedoch mehrere Konkatenationen kombiniert, z.B. ((((BE)I)(SP))((IE)L)). Das sieht hässlich aus; doch glücklicherweise kann man alle Klammern auch weglassen, weil die Reihenfolge, in der Wortteile verknüpft werden, keinen Einfluss auf das resultierende Wort hat. (Dagegen ist die Reihenfolge der Wortteile untereinander entscheidend, wie Punkt 2.5 zeigt).

   Für alle Wörter u, v, w gilt $(uv)w = u(vw)$. Diese Eigenschaft ist auch als Assoziativität bekannt.

   Die Behauptung ergibt sich für $u = \lambda$ nach Punkt 1a (in Verbindung mit der in Punkt 2 eingeführten Schreibweise):

$$(\lambda v)w = vw = \lambda(vw).$$

   Für den Fall $u = xt$ mit $x \in A$ erhält man

$$((xt)v)w \underset{1\mathrm{b}}{=} (x(tv))w \underset{1\mathrm{b}}{=} x((tv)w) \underset{(*)}{=} x(t(vw)) \underset{1\mathrm{b}}{=} (xt)(vw),$$

   wenn die Gültigkeit der Aussage für t vorausgesetzt wird $(*)$. Das ist zulässig, da t um ein Zeichen kürzer ist als u, so dass die Eigenschaft für t als Induktionsvoraussetzung formuliert werden kann.

2.3 Induktionsprinzip

1. Wie in der vorangegangenen Überlegung liefert die rekursive Definition der Wörter ein für viele Situationen brauchbares Induktionsprinzip. Um eine Aussage THEOREM über Wörter zu beweisen, funktioniert oft folgendes Vorgehen:

Induktionsanfang (IA):	Zeige	THEOREM für $v = \lambda$.
Induktionsvoraussetzung (IV):	Nimm	THEOREM für v an.
Induktionsschluss (IS):	Zeige	THEOREM für xv mit $x \in A$.

2. Dieses Prinzip kann benutzt werden, um nachzuweisen, dass λ keinen Einfluss auf die Konkatenation hat. (Vergleiche Punkt 2.2.1a.)

   Für alle Wörter v gilt $v\lambda = v$.

   Denn es gilt:

$$\lambda\lambda \underset{2.2.1}{=} \lambda \text{ und } (xv)\lambda \underset{2.2.1}{=} x(v\lambda) \underset{(*)}{=} xv,$$

   wobei $(*)$ die Anwendung der Induktionsvoraussetzung anzeigt.

   Die oben gezeigte Assoziativität der Konkatenation beruhte bereits auf dem Induktionsprinzip. So lässt sich auch nachweisen, dass die Konkatenation eindeutig ist.

   IA: Die Konkatenation von λ mit einem beliebigen Wort w liefert nach Definition dieses Wort, ist also eindeutig.

   IV: Sei vw für zwei Wörter v, w ein eindeutig bestimmtes Wort.

   IS: Betrachte nun $(xv)w$ für $x \in A$ und Wörter v, w.
   Nach Definition gilt: $(xv)w = x(vw)$. Nach IV ist vw ein bestimmtes Wort, so dass nach den Festlegungen in Punkt 2.1.5 auch $x(vw)$ eindeutig bestimmt ist.

2.4 Gleichheitstest, Länge und Zeichenzählen

1. Die eindeutige Zerlegbarkeit von Wörtern gemäß Punkt 2.1.5 gestattet auch, in einfacher Weise die Gleichheit zweier Wörter rekursiv festzustellen.

   Zwei Wörter v, w sind *gleich* (in Zeichen: $v \equiv w$), wenn sie beide leer sind: $v = \lambda = w$, oder wenn sie beide nicht leer sind sowie $\mathit{head}(v) \equiv \mathit{head}(w)$ und $\mathit{tail}(v) \equiv \mathit{tail}(w)$, wobei ein Gleichheitstest auf dem Alphabet, der ebenfalls mit $\equiv$ bezeichnet wird, vorausgesetzt ist.

2. Ebenfalls rekursiv bestimmt werden kann, wie lang ein Wort ist und wie oft ein bestimmtes Zeichen darin vorkommt:

   (a) $\mathit{length}(\lambda) = 0$
   (b) $\mathit{length}(xv) = \mathit{length}(v) + 1$ für $x \in A, v \in A^*$
   (c) $\mathit{count}(x, \lambda) = 0$
   (d) $\mathit{count}(x, yv) =$ if $x \equiv y$ then $\mathit{count}(x, v) + 1$ else $\mathit{count}(x, v)$
   für $x, y \in A, v \in A^*$.

   Die in (d) verwendete Fallunterscheidung funktioniert wie üblich: Ist die Abfrage wahr, so wird der then-Teil wirksam, sonst der else-Teil.

   Dass durch (a) und (b) eine Abbildung $\mathit{length}\colon A^* \to \mathbb{N}$ definiert wird, lässt sich mit Hilfe des Induktionsprinzips zeigen:

   IA: $\mathit{length}(\lambda) = 0$ ist genau ein Wert aus $\mathbb{N}$ für λ.

   IV: Für ein Wort v sei $\mathit{length}(v)$ genau eine natürliche Zahl.

   IS: Betrachte nun xv für $x \in A$. Nach (b) gilt $\mathit{length}(xv) = \mathit{length}(v) + 1$. Nach IV ist $\mathit{length}(v)$ genau eine natürliche Zahl, so dass der Nachfolger auch genau eine natürliche Zahl ist.

   Analog erweist sich auch $\mathit{count}\colon A \times A^* \to \mathbb{N}$ als Abbildung.

3. Auf dieser Basis lassen sich auch diverse weitere Eigenschaften der eingeführten Operationen zeigen. Als Beispiel wird mit vollständiger Induktion bewiesen, dass die Länge einer Konkatenation gerade die Summe der Längen der Einzelwörter ist, d.h. es gilt für alle $v, w \in A^*$:

$$\mathit{length}(vw) = \mathit{length}(v) + \mathit{length}(w)$$

   IA: $\mathit{length}(\lambda w) = \mathit{length}(w) = 0 + \mathit{length}(w) = \mathit{length}(\lambda) + \mathit{length}(w)$.
   Dabei werden nacheinander die Definition der Konkatenation, eine bekannte arithmetische Eigenschaft und die Definition der Länge ausgenutzt.

   IV: Die Behauptung gelte für v und beliebige w.

   IS: Betrachte av mit $a \in A$ (und beliebiges w):

$$\begin{aligned} \mathit{length}((av)w) &= \mathit{length}(a(vw)) \\ &= \mathit{length}(vw) + 1 \\ &= \mathit{length}(v) + \mathit{length}(w) + 1 \\ &= \mathit{length}(v) + 1 + \mathit{length}(w) \\ &= \mathit{length}(av) + \mathit{length}(w). \end{aligned}$$

Dabei werden nacheinander die Definition der Konkatenation, die Definition der Länge, die Induktionsvoraussetzung, eine arithmetische Eigenschaft und wieder die Definition der Länge ausgenutzt.

2.5 Iterative Darstellung

Die eindeutige Zerlegbarkeit von Wörtern nach Punkt 2.1.5 ergibt zusammen mit dem Induktionsprinzip eine sehr wichtige, gebräuchliche und vertraute Darstellung von Wörtern. Für jedes Wort w existieren eindeutig $n \in \mathbb{N}$ und $x_i \in A$ für $i = 1, \ldots, n$ mit $w = x_1 \cdots x_n$. Das schließt das leere Wort mit ein. In diesem Falle ist $n = 0$, und $x_1 \cdots x_0$ steht für λ. Insgesamt lässt sich also jedes Wort eindeutig in Zeichen als elementare Bausteine zerlegen. Auf den einfachen Beweis wird verzichtet.

2.6 Ansichten von der Menge aller Wörter

Während die vorausgegangenen Informationen über Wörter unverzichtbar für die weiteren Überlegungen sind, dient dieser Abschnitt zur Abrundung. Es wird gezeigt, dass sich die Menge aller Wörter in verschiedenen Formen darstellen lässt. In Punkt 1 wird ein interessanter Zusammenhang zur Algebra und universellen Algebra hergestellt. Punkt 2 charakterisiert A^* durch eine sogenannte Bereichsgleichung. Solche Gleichungen sind in der denotationellen Semantik von Programmiersprachen gebräuchlich. Die Punkte 3 und 4 liefern eine iterative Darstellung von Wörtern, die in der Literatur am häufigsten anzutreffen ist. Die in diesem Kapitel gewählte Einführung wird *axiomatisch* genannt. In Punkt 5 wird die Nähe zu den berühmten Peano-Axiomen der natürlichen Zahlen aufgedeckt. Diese Art des Zugangs eignet sich besonders für den weiteren Umgang mit Wörtern nach Art der funktionalen Programmierung.

1. Die Konkatenation gemäß Punkt 2.2.1 bestimmt eine Abbildung $\cdot : A^* \times A^* \to A^*$, die nach Punkt 2.2.3 assoziativ ist und deshalb A^* zu einem *Monoid* macht mit dem leeren Wort als neutralem Element (vgl. Punkte 2.2.1a und 2.3.2). Da jedes Wort nach Punkt 2.5 eindeutig in “Atome” zerfällt, erweist sich A^* sogar als *freies Monoid.*

2. Die in Punkt 2.1.5 formulierte Zerlegbarkeitseigenschaft der Linksaddition lässt sich explizit dadurch erreichen, dass xv als Paar (x, v) geschrieben wird. Unter Verwendung der Mengenoperationen *disjunkte Vereinigung* $+$ und *kartesisches Produkt* $\times$ erfüllt demnach die Menge A^* aller Wörter über A die Gleichung

$$A^* = \{\lambda\} + (A \times A^*).$$

   A^* ist sogar die kleinste Menge mit dieser Eigenschaft. Deshalb ließe sich diese Mengengleichung auch zur Definition von A^* und damit von Wörtern über A heranziehen.

3. Eine weitere Möglichkeit, A^* zu definieren, die häufig in der Literatur zu finden ist, besteht darin, Wörter direkt als beliebige Tupel von atomaren Zeichen zu wählen:

   (a) λ ist ein Wort,
   (b) für $n > 0$ und $x_i \in A$ $(i = 1, \ldots, n)$ ist $w = x_1 \cdots x_n$ ein Wort.

   In Tupelschreibweise lautet diese Definition:

   (c) das leere Tupel () ist ein Wort,
   (d) das n-Tupel $(x_1, \ldots, x_n)$ mit $n > 0$, $x_i \in A$, $i = 1, \ldots, n$ ist ein Wort.

4. Beachtet man, dass n-Tupel wie in (b) bzw. (d) von Punkt 3 Elemente des n-fachen kartesischen Produkts A^n sind, ergibt sich für die Menge aller Wörter über A folgende Darstellung

$$A^* = \sum_{i=0}^{\infty} A^i,$$

   wobei Σ die disjunkte Vereinigung bezeichnet.

   Korrespondierend zur Linksaddition lassen sich die Potenzen A^i iterieren:

$$A^0 = \{\lambda\} \text{ und } A^{i+1} = A \times A^i \text{ für } \ i \in \mathbb{N}.$$

5. Betrachtet man speziell das einelementige Alphabet $\{|\}$, so entstehen als Wörter Strichfolgen beliebiger Länge, wobei jedes Wort bereits eindeutig durch seine Länge festgelegt ist. Das leere Wort kann also als 0 gesehen werden, und die Linksaddition entspricht der Nachfolgerfunktion natürlicher Zahlen. Die Strichdarstellung natürlicher Zahlen ist als Bierdeckel-Arithmetik bekannt. Spezialisiert man außerdem das Induktionsprinzip für Wörter auf diesen Fall, erhält man ein bekanntes Induktionsprinzip für natürliche Zahlen. Insgesamt erweist sich also der in diesem Kapitel gewählte Zugang zu Wörtern als Verallgemeinerung der durch die Peano-Axiome definierten natürlichen Zahlen.

   Es sei noch folgendes angemerkt. Ergänzte man die Erzeugung von Wörtern explizit um ihre Länge:

   (a) λ ist ein Wort der Länge 0,
   (b) xv ist ein Wort der Länge $n + 1$, falls $x \in A$ und v ein Wort der Länge $n \in \mathbb{N}$ ist,

   so ließe sich das Induktionsprinzip in 2.3 durch vollständige Induktion über die Länge von Wörtern beweisen.