Grundlagen der algebraischen Semantik

Vorlesung an der Fakultät für Mathematik und Informatik der Universität Leipzig [1]

Dr. Rolf Hartwig

zuletzt gehalten im Sommersemester 1997

[1] nach einer vom Lesenden durchgesehenen älteren Mitschrift von Informatikstudenten

Quelle: http://www.informatik.uni-leipzig.de/∼rhartwig/

Der Autor ist jederzeit für inhaltliche Hinweise zur Verbesserung der Vorlesung als auch für Hinweise zu immer noch verbliebenen Druckfehlern dankbar.

Inhaltsverzeichnis

Kapitel 1

Einführung

1.1 Mathematiker und Programmierer

Zur Motivation unserer Beschäftigung mit dem Gegenstand dieser Vorlesung möchte ich eine hypothetische Diskussion zwischen einem Mathematiker und einem Programmierer wiedergeben, die ich so ähnlich in
Blikle, A. *Notes on the mathematical semantics of programming languages.* ICS PAS Reports 445, Warsaw 1981
gefunden habe:

Mathematiker: Was ist ein „Datentyp“?

Programmierer: Datentypen, das sind *integers, reals, arrays, records, stacks, files, characters,* . . .

Mathematiker: Das ist doch keine Antwort auf meine Frage! Ich möchte ganz allgemein wissen, was man unter einem Datentyp versteht.

Programmierer: Warum soll man sich darum kümmern? Genügt es nicht, eine Liste von allen üblichen, gebräuchlichen Datentypen anzugeben?

Mathematiker: Nein, das genügt nicht! Es gibt Programmiersprachen wie ALGOL 68, SIMULA, PASCAL, ADA, MODULA, wo der Nutzer Datentypen nach seinen eigenen Vorstellungen selbst definieren kann. Wenn wir solche Spachen beschreiben, implementieren, lehren und nutzen wollen, müssen wir wissen, *was* ein Datentyp ist.

Programmierer: In solch einem Fall können wir sagen, daß ein Datentyp eine Kollektion, d.h. eine Menge, von allen Objekten eines gegebenen Typs ist; wie z.B. eine Menge von „integers“ oder eine Menge von „real arrays“. So sind Datentypen auch in PASCAL definiert: „Ein Datentyp legt die Menge der Werte fest, die eine Variable annnehmen kann.“ [aus: **Jensen, K., N: Wirth** *PASCAL User Manual.* Berlin-Heidelberg-New York, Springer 1975, p. 12]

Mathematiker: Das stimmt aber nicht für alle Typen. Nehmen wir zum Beispiel *stacks* (Keller) und *queues* (Schlangen). Sie repräsentieren dieselbe Menge, eine Menge von „Ketten“ (Folgen, *strings*), aber sie sind verschieden als Datentypen.

Programmierer: Das ist richtig. Sie unterscheiden sich in ihrem Zugriffsmechanismus. Stacks arbeiten nach dem „first-in-last-out“- und Queues nach dem „first-in-first-out“-Prinzip.

Mathematiker: Das ist der entscheidende Punkt. Um einen Datentyp vollständig zu beschreiben, müssen wir zwei Dinge definieren:

1. die Menge der Objekte eines gegebenen Typs,

2. die Menge der Operationen über diesen Objekten.

Mit anderen Worten, Sie müssen eine Algebra definieren.

Programmierer: Bedeutet das, daß Datentypen Algebren sind?

Mathematiker: Jawohl, aber dies bedarf noch einiger Anmerkungen.

An dieser Stelle verlassen wir das Gespräch zwischen dem Mathematiker und dem Programmierer, um die noch notwendigen Anmerkungen im folgenden selbst zu machen:

1.2 Algebra und Informatik

In der Mathematik versteht man unter einer Algebra (auch universelle Algebra, algebraische Struktur) eine „Trägermenge" mit einer „Operationenfamilie". Um dagegen in der Informatik z.B. den Datentyp *real array* zu definieren, benötigt man die Mengen *real, array* und *integer*, ansonsten könnte man keine Operationen definieren. Das heißt, im allgemeinen genügt in der Informatik der Begriff der traditionellen Algebra nicht mehr, man braucht ein algebraisches Gebilde mit mehreren Trägermengen:

(homogene oder einsortige) Algebra	Heterogene (mehrsortige) Algebra
Gruppen, Ringe, Boolesche Algebren, Verbände	Datentypen, abstrakte Automaten, Syntax und Semantik von Programmiersprachen, aber auch: Vektorräume

Ein anderes Problem der Informatik ist die Partialität vieler Operationen. Dies tritt z.B. beim Selektieren eines Feldelementes A[k] auf. Falls der Wert von k nicht innerhalb der Feldgrenzen liegt, so ist die Selektion undefiniert. Bei arithmetischen Operationen auf realen, d.h. resourcenbeschränkten Rechnern, kann Überlauf oder Unterlauf auftreten. Weitere Undefiniertheiten sind beispielsweise die Division durch Null, das Lesen aus einem leeren Keller oder der Aufruf einer nichtdeklarierten Prozedur usw.

Also ist für die Informatik die Betrachtung *mehrsortiger, partieller Algebren* zweckmäßig. Aber die Theorie partieller Algebren ist sehr kompliziert, deshalb werden hier in der Vorlesung zunächst nur Algebren mit vollen Operationen betrachtet. Dies stellt jedoch keine prinzipielle Einschränkung dar, denn generell ist es möglich, alle partiellen Operationen künstlich zu vervollständigen, in dem jeder „undefinierte" Zustand als Wert „error" der Trägermenge(n) aufgefaßt wird.

1.3 Einführungsbeispiel

Datentyp „Zeichenkeller" (Stacks of characters).

Zur Definition sind mindestens drei Trägermengen erforderlich:

$$\begin{array}{lll}
\underline{\text{Char}} & = \underbrace{\{a, b, c, \ldots, z\}}_{\underline{Char_P}} & \cup \quad \{\underline{char_err}\} \\
\underline{\text{Stack}} & = \underline{Char_P^*} & \cup \quad \{\underline{stack_err}\} \\
\underline{\text{Bool}} & = \{\underline{true}, \underline{false}\} & \cup \quad \{\underline{bool_err}\}
\end{array}$$

Weiterhin sind vier Operationen zweckmäßig:

push	$: \underline{\text{Char}} \times \underline{\text{Stack}} \mapsto \underline{\text{Stack}}$
top	$: \underline{\text{Stack}} \mapsto \underline{\text{Char}}$
pop	$: \underline{\text{Stack}} \mapsto \underline{\text{Stack}}$
empty?	$: \underline{\text{Stack}} \mapsto \underline{\text{Bool}}$

Es bezeichne ε den leeren Keller und sx die Verkettung (Füllung) des Kellers s mit dem Zeichen x. Dann sind die obigen Operationen wie folgt definiert:

$$
\begin{aligned}
\text{push}(x,s) &= sx \\
\text{top}(sx) &= x \\
\text{top}(\varepsilon) &= \underline{char}\_\underline{err} \\
\text{pop}(sx) &= s \\
\text{pop}(\varepsilon) &= \underline{stack}\_\underline{err} \\
\text{empty?}(s) &= \begin{cases} \underline{true} & \text{, falls } s = \varepsilon \\ \underline{bool}\_\underline{err} & \text{, falls } s = \underline{stack}\_\underline{err} \\ \underline{false} & \text{, sonst} \end{cases}
\end{aligned}
$$

Für eine Sprache, die $\underline{\text{Bool}}$enthält, ist es oft sehr nützlich, wenn sie folgende logische Funkionen enthält:

vel	$: \underline{\text{Bool}}^2 \mapsto \underline{\text{Bool}}$
et	$: \underline{\text{Bool}}^2 \mapsto \underline{\text{Bool}}$
non	$: \underline{\text{Bool}} \mapsto \underline{\text{Bool}}$

Weiterhin ist es möglich, eine neue Sorte $\underline{\text{Int}}$ zu dem Datentyp „Zeichenkeller“ zu ergänzen, z.B. um die Kellertiefe zu definieren:

$$\text{depth} : \underline{\text{Stack}} \mapsto \underline{\text{Int}}$$

Eventuell ist es weiter zweckmäßig, das Rechnen über $\underline{\text{Int}}$ hinzuzunehmen.

Wir erhalten so eine Algebra mit vier Trägermengen und verschiedenen Operationen zwischen ihnen. Die Struktur der Algebra, im folgenden *Signatur* genannt, läßt sich z.B. durch das in Abbildung 1 gezeigte Diagramm verdeutlichen.

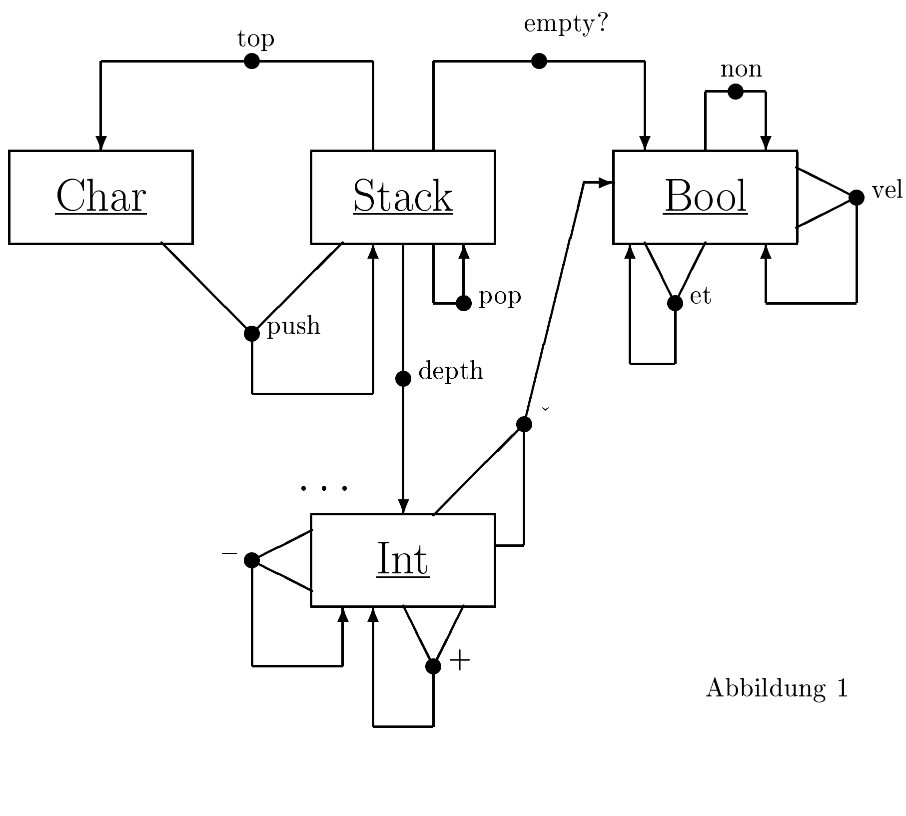

Abbildung 1

Kapitel 2

Heterogene Algebren

2.1 Signatur und Algebra

Definition 2.1

Eine (heterogene) *Signatur* Σ ist ein Tripel $\Sigma = (S, \Omega, \alpha)$, wobei S und Ω Mengen mit $S \neq \emptyset$ und $S \cap \Omega = \emptyset$ sind und α eine Abbildung von Ω in die Menge S^+ aller endlichen nichtleeren Folgen von Elementen aus S.

1. S heißt die Menge der *Sorten(namen)*,
2. Ω die Menge der *Operatoren*, und
3. $\alpha : \Omega \mapsto S^+$ die *Sortigkeit* oder die *Arität*

Es sei $\omega \in \Omega$ mit $\alpha(\omega) = (s_1, s_2, \ldots, s_n, s_{n+1})$. Dann heißt die Folge $w = (s_1, s_2, \ldots, s_n)$ der *Eingang* des Operators ω und wird mit $i_\alpha(\omega)$ bezeichnet. Der *Ausgang* von ω, kurz $o_\alpha(\omega)$ ist dann s_{n+1}. Weiterhin heißt n die *Stelligkeit* des Operators ω.

Bemerkung

Es gilt stets $i_\alpha(\omega) \in S^*$ und $o_\alpha(\omega) \in S$, wobei S^* die Menge aller endlichen Folgen von Elementen aus S (einschließlich der leeren Folge λ) bezeichnet, und wir schreiben deshalb auch: $\alpha : \Omega \mapsto S^* \times S$, d.h. für alle $\omega \in \Omega$ ist $\alpha(\omega) = (w, s)$ mit:

$$w = i_\alpha(\omega) \in S^*$$
$$s = o_\alpha(\omega) \in S$$

Bequeme Schreibweisen

Sei im folgenden $A = (A_s)_{s \in S}$ eine Mengenfamilie.

1. A heißt eine *Teilfamilie* von B ($B = (B_s)_{s \in S}$), falls $A_s \subseteq B_s$ für alle $s \in S$, Kurzform $A \subseteq B$.
2. Analog schreiben wir für zwei Mengenfamilien $A = (A_s)_{s \in S}$ und $B = (B_s)_{s \in S}$
   $C = A \cup B$ bzw. $C = A \cap B$ für $C = (C_s)_{s \in S}$ mit $C_s = A_s \cup B_s$ bzw. $C_s = A_s \cap B_s$ für alle $s \in S$.
3. Die Kurzschreibweise $a \in A$ benutzen wir für $\exists s(s \in S \wedge a \in A_s)$.
4. In Verallgemeinerung des n-fachen Kreuzproduktes definieren wir für $w \in S^*$

$$A^w = \begin{cases} A_{s_1} \times A_{s_2} \times \cdots \times A_{s_n} & \text{falls } w = (s_1, s_2, \ldots, s_n), n \geq 1 \\ \{\emptyset\} & \text{falls } w = \lambda \end{cases}$$

5. Seien A und B wie oben definiert zwei Mengenfamilien. Dann verstehen wir unter einer (eindeutigen) Abbildung h von A in B kurz $h : A \mapsto B$ eine Familie von Abbildungen $h = (h_s)_{s \in S}$ mit $h_s : A_s \mapsto B_s$.
6. Analog zu A^w führen wir für eine Abbildungsfamilie $h = (h_s)_{s \in S}$ auch die Schreibweise h^w bei $w = (s_1, s_2, \dots, s_n)$ ein: Wenn $h : A \mapsto B$, dann $h^w : A^w \mapsto B^w$ vermöge $h^w(a_1, a_2, \dots, a_n) = (h_{s_1}(a_1), h_{s_2}(a_2), \dots, h_{s_n}(a_n))$.

Definition 2.2

Ist $\Sigma = (S, \Omega, \alpha)$ eine Signatur, so heißt $\mathcal{A} = [(A_s)_{s \in S}, (f_\omega)_{\omega \in \Omega}]$ eine *heterogene Algebra* mit der Signatur Σ oder kurz Σ–Algebra, wenn $(A_s)_{s \in S}$ eine Mengenfamilie und $(f_\omega)_{\omega \in \Omega}$ eine Familie von Funktionen bezeichnet, wobei für alle $\omega \in \Omega$ gilt: falls $\alpha(\omega) = (w, s)$, so ist f_ω Funktion von A^w in A_s, d.h. $f_\omega : A^w \mapsto A_s$.

Die nicht notwendig disjunkten Mengen A_s heißen *Trägermengen* von $\mathcal{A}$ (A_s Trägermenge der Sorte s), die f_ω *Operationen* von $\mathcal{A}$ (Grundoperationen). Die nullstelligen Operationen von $\mathcal{A}$ heißen auch *Konstanten*:

$$\alpha(\omega) = (s), \text{also } f_\omega : \{\emptyset\} \mapsto A_s, \text{d.h. } f_\omega(\emptyset) = f_\omega = a \in A_s$$

Somit zeichnet eine Konstante der Sorte s genau ein Element aus A_s aus.

Bemerkung

1. Ist $\mathcal{A} = [(A_s)_{s \in S}; (f_\omega)_{\omega \in \Omega}]$ eine heterogene Algebra, so ist deren Signatur Σ eindeutig bestimmt.
2. Ist in der Signatur $\Sigma = (S, \Omega, \alpha)$ die Sortenmenge S einelementig, also $S = \{s\}$, so reduziert sich der Begriff der heterogenen Σ–Algebra $\mathcal{A}$ zum üblichen Begriff der universellen Algebra (algebraische Struktur, homogene Algebra) :
   (a) es existiert nur eine Trägermenge A
   (b) $\alpha(\omega) = (s, s, \dots, s)$ gibt dann nur noch die Stellenzahl an, ist also äquivalent (begrifflich äquivalent) zu einer Abbildung $\tau : \Omega \mapsto I\!N$
   (c) Damit ist aus Σ nur noch die Angabe von Ω und τ wesentlich, oder einfacher nur die Angabe von $\tau : \Omega \mapsto I\!N$. τ heißt auch *Typ der Algebra*.

Definition 2.3

Algebren gleicher Signatur heißen *ähnlich*.

2.2 Beispiele für Σ–Algebren

1. MEALY–Automat
   Betrachte eine Signatur $\Sigma = (S, \Omega, \alpha)$ mit $S = \{x, y, z\}$, $\Omega = \{1, 2\}$, und $\alpha(1) = (z, x, z)$, $\alpha(2) = (z, x, y)$. Ferner sei der MEALY–Automat $\mathcal{A}$ in üblicher Weise als Quintupel

   $$\mathcal{A} = [X, Y, Z; \delta, \lambda]$$

   gegeben.
   $\mathcal{A}$ läßt sich nun auch als Σ–Algebra auffassen. Dabei hat man:
   $A_x = X$ Eingabealphabet
   $A_y = Y$ Ausgabealphabet
   $A_Z = Z$ Zustandsmenge

$f_1 = \delta : Z \times X \mapsto Z$ Überführungsfunktion
$f_2 = \lambda : Z \times X \mapsto Y$ Ausgabefunktion

2. MOORE–Automat
In analoger Weise kann ein MOORE–Automat

$$\mathcal{A} = [X, Y, Z; \delta, \mu]$$

mit Eingabealphabet X, Ausgabealphabet Y und Zustandsmenge Z sowie Überführungsfunktion δ und Ausgabefunktion μ als eine Σ–Algebra aufgefaßt werden mit z.B. der Signatur $\Sigma = (S, \Omega, \alpha)$, wobei $S = \{x, y, z\}$, $\Omega = \{\delta, \mu\}$, und $\alpha(\delta) = (z, x, z)$, $\alpha(\mu) = (z, y)$ ist.

3. ROBIN SCOTT–Automat (initialer Automat)
Automatenbeschreibung :

$$\mathcal{A} = [Z, X, z_0, F, \delta]$$

mit Überführungsfunktion $\delta : Z \times X \mapsto Z$, Anfangszustand $z_0 \in Z$ und $F \subseteq Z$ als Menge der Endzustände. Dieser Automat kann auch als heterogene Algebra $[(A_s)_{s\in S}; (f_\omega)_{\omega\in\Omega}]$ aufgefaßt werden mit:

$S = \{z, x\}$
$\Omega = \{w_0, w_\delta\} \cup \{w_t \mid t \in F\}$
$\alpha(\omega_0) = (z),\ \alpha(\omega_t) = (z)$ für alle $t \in F$
$\alpha(\omega_\delta) = (z, x, z)$
$A_z = Z, A_x = X$
$f_{\omega_0} = z_0,\ f_{\omega_t} = t$ für alle $t \in F$ (t Konstanten)
$f_{\omega_\delta} = \delta.$

4. Einfache Beispiele für homogene Algebren:

   (a) $[\mathrm{Pow}(M); \cup, \cap]$,
   Potenzmenge einer Menge M mit Vereinigung und Durchschnitt,

   (b) $[\{W, F\}; \wedge, \vee]$,
   die Zweiermenge der Wahrheitswerte mit deren bekannten Verknüpfungen sowie

   (c) $[I\!N; +, *]$,
   die Menge der natürlichen Zahlen mit Addition und Multiplikation

   sind homogene Algebren vom Typ(2,2).

Nun noch eine bekannte homogene Algebra vom Typ (0,1), die PEANO–Algebra:

$$[I\!N, 0,']$$,

mit der Menge $I\!N$ der natürlichen Zahlen als Trägermenge und der Konstanten 0 und der Nachfolgerfunktion ' als deren Operationen.

5. [<u>AUS</u> ; neg , con , alt]
Dabei ist <u>AUS</u> die Menge aller pfeilfreien aussagenlogischen Ausdrücke, neg die syntaktische Operation der Bildung negierter Ausdrücke und con bzw. alt sind die syntaktischen Operationen der Bildung von Konjunktion und Alternative.
Dies ist ein Beispiel für eine homogene Algebra vom Typ (1,2,2) mit solch bemerkenswerten Eigenschaften, das es uns auch später noch interessieren wird, wenn wir sog. Termalgebren untersuchen.
Bei dem üblichen aussagenlogischen Aufbau würde definiert werden für $H_1, H_2, H \in$<u>AUS</u>:

neg(H) $= \neg$H
con(H_1, H_2) $= (H_1 \wedge H_2$
alt(H_1, H_2) $= (H_1 \vee H_2)$

6. $\underline{S} = [W, X, ⚲, \circ]$
   sei ein sogenanntes „semiotisches Quadrupel“, das heißt X ist ein Alphabet (im Sinne der Theorie der Zeichenreihen, der sog. Semiotik), W sei dann die Wortmenge über X, ⚲ bezeichnet das leere Wort und $\circ$ die Operation der Verkettung eines Wortes mit einem Zeichen.
   Dies ist eine heterogene Algebra der Signatur $\Sigma = (S, \Omega, \alpha)$ mit

   $$S = \{w, z\}$$
   $$\Omega = \{\varepsilon, \kappa\}$$
   $$\alpha(\varepsilon) = (w)$$
   $$\alpha(\kappa) = (w, z, w)$$
   $$A_w = W,\ A_z = X$$
   $f_\varepsilon =$ ⚲ Konstante der Sorte w
   $$f_\kappa = \circ : W \times X \mapsto W$$

7. *Die Syntax einer Programmiersprache*
   stellt stets eine heterogene Algebra dar. Dies wird am **Beispiel** einer sehr einfachen Sprache erläutert:

   - <Ide> - die Menge der Identifier sei irgendwie gegeben.

   - Ausdrücke(Expressions):
     <Exp> ::= $\underline{0}$ | $\underline{1}$ | $\underline{\mathit{true}}$ | $\underline{\mathit{false}}$ | <Ide> | ¬ <Exp> |
     (<Exp> = <Exp>) | (<Exp> + <Exp>)

   - Anweisungen(Commands):
     <Com> ::= <Ide> := <Exp> | $\underline{\mathit{if}}$ <Exp> $\underline{\mathit{then}}$ <Com> $\underline{\mathit{else}}$ <Com> $\underline{\mathit{fi}}$ |
     <Com>; <Com>

   Damit haben wir eine Algebra mit drei Trägermengen (also eine dreisortige Algebra):

   $$\text{Ide}, \text{Exp}, \text{Com}$$

   und zu jeder Regel der kontextfreien Grammatik, die der obigen BACKUS-NAUR–Form (BNF) entspricht, und zu jedem Identifier genau eine Operation.
   Die Signatur dieser Algebra kann in leicht verständlicher Weise aus folgenden Angaben abgelesen werden:

$$
\begin{array}{lll}
0 & : & \mapsto \text{Exp} \\
1 & : & \mapsto \text{Exp} \\
\mathit{true} & : & \mapsto \text{Exp} \\
\mathit{false} & : & \mapsto \text{Exp} \\
I & : & \mapsto \text{Exp} \quad \text{für alle } I \in \text{Ide} \\
\neg & : & \text{Exp} \mapsto \text{Exp} \\
= & : & \text{Exp} \times \text{Exp} \mapsto \text{Exp} \\
+ & : & \text{Exp} \times \text{Exp} \mapsto \text{Exp} \\
:= & : & \text{Ide} \times \text{Exp} \mapsto \text{Com} \\
\mathit{if} & : & \text{Exp} \times \text{Com} \times \text{Com} \mapsto \text{Com} \\
; & : & \text{Com} \times \text{Com} \mapsto \text{Com}
\end{array}
$$

Die zugeordneten syntaktischen Operationen der Algebra lassen sich ebenfalls aus der BNF ablesen:

f_0	$= \underline{0}$	(Konstante)
f_1	$= \underline{1}$	(Konstante)
f_{true}	$= \underline{true}$	(Konstante)
f_{false}	$= \underline{false}$	(Konstante)
f_I	$= I$	für alle $I \in$ Ide (Konstanten)
$f_{\neg}(E)$	$= \neg E$	für $E \in$ Exp,
$f_{=}(E_1, E_2)$	$= (E_1 = E_2)$	für $E_1, E_2 \in$ Exp,
$f_{+}(E_1, E_2)$	$= (E_1 + E_2)$	für $E_1, E_2 \in$ Exp,
$f_{:=}(I, E)$	$= I := E$	für $I \in$ Ide, $E \in$ Exp,
$f_{if}(E, C_1, C_2)$	$= \underline{if} E \underline{then} C_1 \underline{else} C_2 \underline{fi}$	für $E \in$ Exp, $C_1, C_2 \in$ Com
$f_{;}(C_1, C_2)$	$= C_1; C_2$	für $C_1, C_2 \in$ Com

Kapitel 3

Algebraische Hülle und Homomorphiebegriff

3.1 Algebraische Hülle, Erzeugendensystem, algebraische Induktion

Allgemeine Bemerkung zu Bezeichnungen

Im folgenden wollen wir eine Konvention zu den am häufigsten auftretenden Bezeichnungen treffen, die uns viel Schreibarbeit ersparen soll.
Wenn nicht ausdrücklich anders angegeben ist, so werden stillschweigend die Bezeichnungen der Signatur Σ und einer Σ–Algebra $\mathcal{A}$ stets so gewählt wie in den ersten Definitionen angegeben. Das heißt, daß stets

$$\Sigma = (S, \Omega, \alpha)$$

angenommen wird, und daß immer dann, wenn ohne weitere Voraussetzungen von einer Algebra $\mathcal{A}$ gesprochen wird, diese eine Σ–Algebra ist mit

$$\mathcal{A} = [A, F], \qquad \text{wobei} \quad A = (A_s)_{s \in S} \quad \text{und} \quad F = (f_s)_{\omega \in \Omega}$$

zu setzen ist.

Definition 3.1

$\mathcal{A} = [A, F]$ sei Σ-Algebra. Dann heißt die Teilfamilie $B \subseteq A$ *abgeschlossen bezüglich* der Operation f_ω aus F gdw. mit $(a_1, a_2, \ldots, a_n) \in B^{i_\alpha(\omega)}$ stets auch $f_\omega(a_1, a_2, \ldots, a_n) \in B_{o_\alpha(\omega)}$ gilt. B heißt *abgeschlossen gegenüber* F bzw. *abgeschlossen in* $\mathcal{A}$ genau dann, wenn B abgeschlossen bezüglich jeder Operation aus F ist.
Die Teilfamilie $B \subseteq A$ heißt *algebraische Hülle* der Teilfamilie $C \subseteq A$, symbolisch $B = [C]_{\mathcal{A}}$ oder $B = [C]$, genau dann, wenn

$$B = \bigcap \left\{ A^{'} \mid C \subseteq A^{'} \subseteq A \text{ und } A^{'} \text{ abgeschlossen in } \mathcal{A} \right\}.$$

Bemerkung

1. Die algebraische Hülle von C ist die *kleinste* (bezüglich $\subseteq$) abgeschlossene Teilfamilie, die C umfaßt.
2. Ist C abgeschlossen gegenüber F, so $[C]_{\mathcal{A}} = C$.

Satz 3.1 (Hülleneigenschaften)

Für die algebraische Hülle gelten die Hülleneigenschaften:

1. Einbettung: $B \subseteq [B]_{\mathcal{A}}$.
2. Monotonie: *Wenn* $B \subseteq C$, *so* $[B]_{\mathcal{A}} \subseteq [C]_{\mathcal{A}}$.
3. Abgeschlossenheit: $[[B]_{\mathcal{A}}]_{\mathcal{A}} = [B]_{\mathcal{A}}$.

Beweis Der Beweis ist trivial wegen der Eigenschaften von $\bigcap$.

Satz 3.2 (Darstellung der algebraischen Hülle $[C]_{\mathcal{A}}$)

Setzt man $K = (K_s)_{s\in S}$ *mit* $K_s = \{f_\omega \mid \omega \in \Omega \wedge \alpha_\omega = (s)\}$ *als die Familie der Konstanten der Sorte* s, *und weiter*

$$C^{(0)} = C \cup K$$

sowie für alle $i > 0$

$$C^{i+1} = C^{(i)} \cup (\{f_\omega(a_1,\ldots,a_n) \mid \omega \in \Omega \wedge n \geq 1 \wedge \alpha(\omega) = (w,s) \wedge (a_1,\ldots,a_n) \in (C^{(i)})^w\})_{s\in S},$$

so gilt

$$[C]_{\mathcal{A}} = \bigcup_{i\in I\!N} C^{(i)}$$

Beweis

Es sei $B = \bigcup_{i\in I\!N} C^{(i)}$. Klar ist sofort, daß $C \subseteq B \subseteq A$ gilt. Wir zeigen noch

1. B ist abgeschlossen in $\mathcal{A}$. Damit gilt, daß $[C]_{\mathcal{A}} \subseteq B$.
2. $B \subseteq [C]_{\mathcal{A}}$.

1. Sei $\omega \in \Omega$ beliebig und $\alpha(\omega) = (w,s) = (s_1, s_2, \ldots, s_n, s)$ und sei $(a_1, a_2, \ldots, a_n) \in B^w$ beliebig gewählt. Dann existieren $i_1, i_2, \ldots, i_n$ mit $a_j \in C_{s_j}^{(i_j)}$ für $i \leq j \leq n$ (entsprechend der Bildung von B). Sei nun $i = \max\{i_1, i_2, \ldots, i_n\}$, dann ist nach Definition der $C^{(i)}$: $a_j \in C_{s_j}^{(i)}$ für alle $1 \leq j \leq n$. Das heißt, daß $(a_1, a_2, \ldots, a_n) \in (C_{s_1}^{(i)} \times C_{s_2}^{(i)} \times \cdots \times C_{s_n}^{(i)})$, also $(a_1, a_2, \ldots, a_n) \in (C^{(i)})^w$. Damit ist nach Definition der $C^{(i)}$:
$f_\omega(a_1, a_2, \ldots, a_n) \in C_s^{(i+1)} \subseteq B_s$.
2. Es gilt für alle $i : C^{(i)} \subseteq [C]_{\mathcal{A}}$.
Beweis durch Induktion:
(IA) $C^{(0)} = C \cup K$ (siehe Definition). $C \subseteq [C]_{\mathcal{A}}$ wegen Hülleneigenschaften. Außerdem ist $[C]_{\mathcal{A}}$ abgeschlossen gegenüber allen Operationen aus $\mathcal{A}$, speziell also auch bezüglich aller nullstelligen Operationen aus $\mathcal{A}$. Daraus folgt $K \subseteq [C]_{\mathcal{A}}$, und somit auch $C^{(0)} \subseteq [C]_{\mathcal{A}}$
(IS) Sei $C^{(i)} \subseteq [C]_{\mathcal{A}}$. Da $[C]_{\mathcal{A}}$ abgeschlossen gegenüber allen Operationen aus $\mathcal{A}$ ist, gilt nach Definition auch $C^{(i+1)} \subseteq [C]_{\mathcal{A}}$.
Damit haben wir auch für die Vereinigung $B = \bigcup_{i\in I\!N} C^{(i)} \subseteq [C]_{\mathcal{A}}$.

q.e.d.

Satz 3.3 (über die Hülle von $\emptyset$)

Sei $\mathcal{A} = [(A_s)_{s\in S}; (f_\omega)_{\omega\in\Omega}]$ Σ*-Algebra und sei* $\underline{\emptyset} = (\emptyset)_{s\in S}$ *die* S*-sortige Familie leerer Mengen.*

1. $[\underline{\emptyset}]_{\mathcal{A}} = [K]_{\mathcal{A}}$, *falls* $K = (K_s)_{s\in S}$ *mit* $K_s = \{c \mid \exists\omega(\omega \in \Omega \wedge \alpha(\omega) = (s) \wedge f_\omega(\lambda) = f_\omega = c)\}$ *ist die Menge der Konstanten der Sorte* s.

2. $[\underline{\emptyset}]_{\mathcal{A}} = \underline{\emptyset}$ *gdw. es in* Ω *keinen nullstelligen Operator gibt.*

Beweis Der Beweis ist trivial, vergl. vorherigen Beweis, Teil 2. (IA).

Definition 3.2

Es seien $\mathcal{A} = [A; (f_\omega)_{\omega\in\Omega}]$ und $\mathcal{B} = [B, (f'_\omega)_{\omega\in\Omega}]$ zwei Σ-Algebren. Dann heißt $\mathcal{B}$ eine *Unteralgebra* von $\mathcal{A}$ gdw. $B \subseteq A$ und für alle $(a_1, a_2, \ldots, a_n) \in B^w$ gilt: $f'_\omega(a_1, a_2, \ldots, a_n) = f_\omega(a_1, \ldots, a_n)$.

Bemerkung

1. $\mathcal{A}$ sei Σ–Algebra. Dann ist $\mathcal{B} = [B, (f'_\omega)_{\omega\Omega}]$ Unteralgebra von $\mathcal{A}$ gdw:
   (a) $B \subseteq A$,
   (b) für alle $\omega \in \Omega$ ist $f'_\omega = f_\omega|_{B^{i_\alpha(\omega)}}$ (Einschränkung),
   (c) B abgeschlossen gegenüber $(f'_\omega)_{\omega\in\Omega}$

   ist.
2. Zu jedem $B \subseteq A$ gibt es damit höchstens eine Unteralgebra $\mathcal{B}$ von $\mathcal{A}$; diese ist durch B (und natürlich $\mathcal{A}$) eindeutig festgelegt. Deshalb schreibt man oft auch: $\mathcal{B} = [B; (f_\omega)_{\omega\in\Omega}]$ statt $\mathcal{B} = [B, (f'_\omega)_{\omega\in\Omega}]$.

Satz 3.4

Es sei $\mathcal{A}$ *eine* Σ*–Algebra und* $X \subseteq A$*. Dann ist* $\mathcal{B} = [[X]_{\mathcal{A}}, (f_\omega)_{\omega\in\Omega}]$ *Unteralgebra von* $\mathcal{A}$*.*

Beweis Der Beweis ist nach dem vorangegangenen trivial.

Definition 3.3

Man nennt $\mathcal{B} = [[X]_{\mathcal{A}}, F]$ die *von der Familie* X *in* $\mathcal{A}$ *erzeugte Unteralgebra*; die Mengenfamilie X heißt dann ein *Erzeugendensystem* von $\mathcal{B}$.

Bemerkung

1. Eine Algebra kann auch ein leeres Erzeugendensystem besitzen
   (s. Satz über Hülle von $\underline{\emptyset}$)
2. X ist genau dann Erzeugendensystem der Algebra $\mathcal{A} = [A, F]$, wenn $[X]_{\mathcal{A}} = A$ ist.

Beispiele für Erzeugendensysteme

1. $\mathcal{A} = [\underline{AUS}\,;\, neg\,,\, con\,,\, alt]$ die Algebra der pfeilfreien aussagenlogischen Ausdrücke (vgl Bsp. 5 vorn) besitzt als Erzeugendensystem die Menge der aussagenlogischen Variablen V, denn es ist $[V]_{\mathcal{A}} = \underline{AUS}$. Dies erkennt man leicht, wenn man die üblichen Ausdrucksdefinitionen vergleicht mit der Bildung der algebraischen Hülle. Die Ausdrücke entstehen gerade dadurch, daß man ausgehend von den Variablen immer wieder die oben angeführten syntaktischen Funktionen anwendet.

2. Vergleiche voriges Beispiel, Punkt 6. Das „semiotische Quadrupel“ S=[W,X; ⚲ ,∘] besitzt als Erzeugendensystem das Paar $(\emptyset, X)$. Denn zunächst gilt, daß ⚲ $\in W$, weil ⚲ Konstante der Sorte w ist. Damit wird $X \subseteq W$ wegen ⚲ $\circ\, x = x \in W$ für alle $x \in X$. (Abschluß gegenüber $\circ$ gefordert) und wegen $o_\alpha(\kappa) = w$. Über weitere Verkettungen werden damit alle Wörter über X erzeugt.
3. $\mathcal{G} = [\underline{G}, 0, +]$, die Algebra der ganzen Zahlen mit der Konstanten 0 und der Addition +, besitzt als Erzeugendensystem $\{-1, 1\}$ (und jede umfassende Menge).
   Zum Begriff der Unteralgebra betrachte man $\mathcal{N} = [I\!N, 0, +]$ die Algebra der natürlichen Zahlen. Sie stellt eine Unteralgebra von $\mathcal{G}$ dar. $\mathcal{G}$ ist bekanntermaßen eine Gruppe, d.h. aber natürlich <u>nicht</u>, daß die Unteralgebra $\mathcal{N}$ auch Untergruppe von $\mathcal{G}$ ist!

Satz 3.5 (Prinzip der algebraischen Induktion)

Es sei $\mathcal{A} = [A, F]$ eine Σ–Algebra und X ein Erzeugendensystem von $\mathcal{A}$. Wenn dann für eine Teilfamilie B von A folgendes gilt:

1. *$X \subseteq B$*
2. *B ist abgeschlossen gegenüber allen Operationen der Algebra (also gegenüber F),*

so ist $B = A$.

Beweis

Nach Voraussetzung ist $B \subseteq A$. Wegen 1. ist $X \subseteq B$, also wegen der Monotonie ist $[X]_\mathcal{A} \subseteq [B]_\mathcal{A}$. Wegen 2. ist $[B]_\mathcal{A} = B$ und nach Voraussetzung ist $[X]_\mathcal{A} = A$, also $A \subseteq B$ und damit $A = B$. q.e.d.

Bemerkung

Damit ist ein Beweisverfahren, die algebraische Induktion gegeben, das wie folgt anzuwenden ist: Es sei $H(a)$ für alle $a \in A$ zu zeigen. X sei Erzeugendensystem von $\mathcal{A}$. Dann sind zwei Beweisschritte auszuführen:

(IA) Zeige $H(x)$ für alle $x \in X$
(IS) Zeige für beliebige $\omega \in \Omega$ (sei $\alpha(\omega) = (w, s)$ angenommen) und beliebige $(a_1, a_2, \ldots, a_n) \in A^w$: Wenn $H(a_1) \wedge H(a_2) \wedge \cdots \wedge H(a_n)$, so auch $H(f_\omega(a_1, a_2, \ldots, a_n))$.

Die Rechtfertigung für diesen Sachverhalt folgt aus obigem Satz, denn man setze
$B_s = \{a \mid a \in A_s \wedge H(a)\}$ und $B = (B_s)_{s \in S}$.

3.2 Homomorphiebegriff

Definition 3.4

$\mathcal{A} = [A, (f_\omega)_{\omega\in\Omega}]$ und $\mathcal{B} = [B, (g_\omega)_{\omega\in\Omega}]$ seien zwei Σ–Algebren. Ein Σ–*Homomorphismus* (oder kurz Homomorphismus) von $\mathcal{A}$ in $\mathcal{B}$ ist eine eindeutige Abbildung(sfamilie) $h : A \mapsto B$, so daß für alle $\omega \in \Omega$ mit $\alpha(\omega) = (s_1, s_2, \ldots, s_n, s)$ und für alle $(a_1, a_2, \ldots, a_n) \in A_{s_1} \times A_{s_2} \times \cdots \times A_{s_n}$ gilt:

$$h_s f_\omega(a_1, \ldots, a_n) = g_\omega(h_{s_1} a_1, \ldots, h_{s_n} a_n)$$

$$\begin{array}{ccc} A^w & \xrightarrow{f_\omega} & A_s \\ h^w \downarrow & & \downarrow h_s \\ B^w & \xrightarrow{g_\omega} & B_s \end{array}$$

Spezialfälle

1. Ein eineindeutiger Homomorphismus von $\mathcal{A}$ in $\mathcal{B}$ heißt *Einbettung* oder *Injektion*.
2. Ein eineindeutiger Homomorphismus von $\mathcal{A}$ auf $\mathcal{B}$ heißt *Isomorphismus* oder *Bijektion*. $\mathcal{A}$ und $\mathcal{B}$ heißen dann *isomorph*, in Zeichen: $\mathcal{A} \cong \mathcal{B}$
3. Ein Homomorphismus von $\mathcal{A}$ auf $\mathcal{B}$ heißt *Epimorphismus* oder *Surjektion*. $\mathcal{B}$ heißt dann *homomorphes Bild* von $\mathcal{A}$.
4. Ein Homomorphismus von $\mathcal{A}$ in $\mathcal{A}$ heißt *Endomorphismus* von $\mathcal{A}$

Folgerung 3.6

(aus dem Homomorphiebegriff)
Für alle $\omega \in \Omega$ mit $i_\alpha(\omega) = \lambda$, d.h. alle nullstelligen Operatoren (Konstanten), $o_\alpha(\omega) = s$ folgt $h_s f_\omega = g_\omega$, d.h. die Konstanten von $\mathcal{A}$ werden auf die Konstanten von $\mathcal{B}$ abgebildet.

Satz 3.7 (Nacheinanderausführung von Homomorphismen)

$\mathcal{A}, \mathcal{B}, \mathcal{C}$ seien Σ–Algebren, h sei ein Homomorphismus von $\mathcal{A}$ in $\mathcal{B}$, g sei Homomorphismus von $\mathcal{B}$ in $\mathcal{C}$. Dann ist $g \circ h$ Homomorphismus von $\mathcal{A}$ in $\mathcal{C}$.

Beweis

1. $g \circ h : A \mapsto C$ als Abbildung klar

2. $$\begin{aligned} (g \circ h)_s(f_\omega^A(a_1, a_2, \ldots, a_n)) &= g_s(h_s f_\omega^A(a_1, \ldots, a_n)) \\ &= g_s f_\omega^B(h_{s_1} a_1, \ldots, h_{s_n} a_n) \\ &= f_\omega^C(g_{s_1} h_{s_1} a_1, \ldots, g_{s_n} h_{s_n} a_n) \\ &= f_\omega^C((g \circ h)_{s_1}(a_1), \ldots, (g \circ h)_{s_n}(a_n)) \end{aligned}$$ q.e.d.

Lemma 3.8

Sei $h : A \mapsto B$ ein Σ–Homomorphismus von $\mathcal{A}$ in $\mathcal{B}$. Weiter sei $X \subseteq A$. Dann ist

$$h([X]_\mathcal{A}) = [h(X)]_\mathcal{B}$$

.

Beweis

Der Beweis kann wegen des Darstellungssatzes für $[X]_{\mathcal{A}}$ mit vollständiger Induktion unter Verwendung der Homomorphieeigenschaft (Operationstreue) geführt werden. Man betrachte die Ausführung als eine Übungsaufgabe.

Satz 3.9 (über die Eindeutigkeit der homomorphen Fortsetzung)

Es seien $\mathcal{A} = [A, F]$ und $\mathcal{B} = [B, G]$ zwei Σ–Algebren. X sei Erzeugendensystem von $\mathcal{A}$ und φ_0 eine eindeutige Abbildungsfamilie von X in B. Dann gibt es höchstens einen Σ–Homomorphismus φ von $\mathcal{A}$ in $\mathcal{B}$, der die gegebene Abbildung φ_0 fortsetzt.

Beweis

Zum Beweis verwenden wir die algebraische Induktion.
Angenommen, es existieren zwei Homomorphismen φ und ψ von $\mathcal{A}$ in $\mathcal{B}$, die φ_0 fortsetzen:

$$\varphi\,|_X = \psi\,|_X = \varphi_0$$

Wir betrachten nun die Mengenfamilie $M = (M_s)_{s \in S}$ mit $M_s = \{a \mid a \in A_s \wedge \varphi_s(a) = \psi_s(a)\}$, also Mengen bildgleicher Elemente von A. Dann gilt:

1. $X_s \subseteq M_s \subseteq A_s$ für alle $s \in S$ nach Voraussetzung
2. M ist abgeschlossen gegenüber F:
   Sei $\omega \in \Omega$ mit $\alpha(\omega) = (w, s)$ und es sei $(a_1, a_2, \ldots, a_n) \in M^w$. Dann $\varphi_{s_i}(a_i) = \psi_{s_i}(a_i)$ für $1 \leq i \leq n$, falls $w = (s_1, \ldots, s_n)$. Also gilt, da die Abbildungen φ und ψ Homomorphismen sind:
   $$\begin{aligned} \varphi_s(f_\omega(a_1, \ldots, a_n)) &= g_\omega(\varphi_{s_1} a_1, \ldots, \varphi_{s_n} a_n) \\ &= g_\omega(\psi_{s_1} a_1, \ldots, \psi_{s_n} a_n) \quad (\text{da } a_i \in M_{s_i}) \\ &= \psi_s f_\omega(a_1, \ldots, a_n) \end{aligned}$$
   d.h. auch $f\omega(a_1, \ldots, a_n) \in M_s$. Damit ist nach dem Prinzip der algebraischen Induktion $M = A$. Also ist $\varphi = \psi$.

q.e.d.

Kapitel 4

Termalgebren

4.1 Freie Algebren

Definition 4.1

Die Σ–Algebra $\mathcal{A} = [A, F]$ heißt *frei* mit dem freien Erzeugendensystem X (über der Klasse aller Σ–Algebren) genau dann, wenn X Erzeugendensystem von $\mathcal{A}$ ist, so daß für jede Σ–Algebra $\mathcal{B} = [B, G]$ und jede eindeutige Abbildungsfamilie φ_0 von X in $\mathcal{B}$ genau ein Homomorphismus von $\mathcal{A}$ in $\mathcal{B}$ existiert, der φ_0 fortsetzt.

$$\begin{tikzcd} \mathcal{A} \arrow[dr, "\varphi"] & \\ & \mathcal{B} \\ X \arrow[uu, "\subseteq"] \arrow[ur, "\varphi_0"'] & \end{tikzcd}$$

Satz 4.1 (Isomorphie freier Algebren)

$\mathcal{A} = [A, F], \mathcal{B} = [B, G]$ seien zwei freie Σ–Algebren mit den freien Erzeugendensystemen X bzw. Y. Dann gilt: $\mathcal{A}$ und $\mathcal{B}$ sind isomorph, wenn ihre freien Erzeugendensysteme X, Y sortenweise gleichmächtig sind.

Beweis

Seien X und Y sortenweise gleichmächtig. Also existiert eine eineindeutige Abbildung ψ von X auf Y (nach Definition der Gleichmächtigkeit). ψ ist Abbildung von X in B. Da $\mathcal{A}$ frei über X ist, existiert genau ein Homomorphismus $\overline{\psi}$ von $\mathcal{A}$ in $\mathcal{B}$ mit $\overline{\psi}\,|_X = \psi$. Nach Voraussetzung ist $\overline{\psi}(A) = \overline{\psi}([X]_{\mathcal{A}})$. Nach obigem Hilfsatz folgt $\overline{\psi}(A) = [\overline{\psi}(X)]_{\mathcal{B}} = [\psi(X)_{\mathcal{B}}] = [Y]_{\mathcal{B}} = B$.Also ist $\overline{\psi}$ Homomorphismus von A *auf* B.
Analog erhält man durch Vertauschen von $\mathcal{A}$ und $\mathcal{B}$, daß es genau einen Homomorphismus $\overline{\psi}^{-1}$ von $\mathcal{B}$ *auf* $\mathcal{A}$ gibt, so daß: $\overline{\psi}^{-1}\,|_Y = \psi^{-1}$ ist.
Nun gilt (wie im folgenden gezeigt wird):

$$\overline{\psi} \circ \overline{\psi}^{-1} = \iota_B \text{ und } \overline{\psi}^{-1} \circ \overline{\psi} = \iota_A.$$

Das bedeutet, daß $\overline{\psi}^{-1} = \overline{\psi}^{-1}$ ist, also $\overline{\psi}$ Isomorphismus von $\mathcal{A}$ auf $\mathcal{B}$ ist.

Dazu betrachte man $\psi^{-1} \circ \overline{\psi}^{-1}$. Als Nacheinanderausführung zweier Homomorphismen ist dies wieder Homomorphismus, und zwar von $\mathcal{A}$ in $\mathcal{A}$. Es ist weiter für alle $s \in S$ und beliebige $x \in X_s$:

$$\begin{aligned}(\overline{\psi^{-1}} \circ \overline{\psi})(x) &= \overline{\psi_s}^{-1}(\overline{\psi_s}(x)) \\ &= \overline{\psi_s}^{-1}(\psi_s(x)) \\ &= \overline{\psi_s}^{-1}(y) \quad \text{mit } y = \psi_s(x),\ x = \psi_s^{-1}(y)\end{aligned}$$

Also ist $(\overline{\psi^{-1}} \circ \overline{\psi}) \mid_X = \iota_X$. Da $\mathcal{A}$ frei über X ist, existiert genau ein Homomorphismus von $\mathcal{A}$ in $\mathcal{A}$, der ι_X fortsetzt, das ist — weil ι_A Homomorphismus ist ! — also ι_A. Folglich (weil auch $\overline{\psi^{-1}} \circ \overline{\psi}$ Homomorphismus von $\mathcal{A}$ in $\mathcal{A}$ ist).

$$\overline{\psi^{-1}} \circ \psi = \iota_A.$$

Analog $\overline{\psi} \circ \overline{\psi^{-1}} = \iota_B.$ q.e.d.

4.2 PEANO–Algebra (analytische Syntax)

Definition 4.2

Eine Σ–Algebra $\mathcal{E} = [(E_s)_{s \in S}, (f_\omega)_{\omega \in \Omega}]$ heißt Σ–*Ausdrucksalgebra (Termalgebra)* über dem *Alphabet* X oder auch Σ–PEANO-*Algebra* über der PEANO–*Basis* X, genau dann, wenn $X = (X_s)_{s \in S}$ und für alle $s \in S$ $X_s \subseteq E_s$ ist, und die folgenden Bedingungen

(P1) $\forall \omega \forall s \forall \mathsf{a}(\omega \in \Omega \wedge o_\alpha(\omega) = s \wedge \mathsf{a} \in E^{i_\alpha(\omega)} \rightarrow f_\omega(\mathsf{a}) \notin X)$

(P2) $\forall \omega_1 \forall \omega_2 \forall s \forall \mathsf{a} \forall \mathsf{b}(\omega_1, \omega_2 \in \Omega \wedge \mathsf{a} \in E^{i_\alpha(\omega_1)} \wedge \mathsf{b} \in E^{i_\alpha(\omega_2)} \wedge o_\alpha(\omega_1) = o_\alpha(\omega_2) = s \wedge f_{\omega_1}(\mathsf{a}) = f_{\omega_2}(\mathsf{b}) \rightarrow \omega_1 = \omega_2 \wedge \mathsf{a} = \mathsf{b})$

(P3) $[X]_\mathcal{B} = E$

gelten. Die Elemente von E_s heißen *Terme* oder *Ausdrücke* der Sorte s, die von X_s heißen *Variablen* der Sorte s.

Bemerkung zur Bezeichnung PEANO–**Algebra**

Der Begriff PEANO–Algebra wird verwendet, weil die Forderungen (P1),(P2),(P3) verallgemeinerte PEANO–Axiome darstellen.
Um sich dies zu veranschaulichen, betrachte man den Spezialfall $S = \{z\}, \Omega = \{'\}, \alpha(') = (z,z),\ f'$: Nachfolgerbildung im Bereich der natürlichen Zahlen, also $f'(a) = a'$ für natürliche Zahlen a, und als Erzeugendensystem wähle $X = \{0\}$.
Nun bezeichnen wir die dann einzige Trägermenge $E_z = E$ der sich ergebenden homogenen Algebra mit $I\!N$. Dann gehen unsere Axiome (P1),(P2),(P3) in die bekannten PEANO–Axiome für die natürlichen Zahlen über:

(P1) $\forall a(a \in I\!N \rightarrow a' \neq 0)$,
in Worten: Null ist nicht Nachfolger irgendeiner Zahl.

(P2) $\forall a \forall b(a, b \in I\!N \wedge a' = b' \rightarrow a = b)$
in Worten: Jede Zahl ist Nachfolger höchstens einer Zahl.

(P3) Zur Überführung der relativ komplexen Bedingung (P3) in die restlichen drei bekannten PEANO–Axiome sei hier nochmal die Erklärung der algebraischen Hülle eingefügt:

1. $X \subseteq E$
2. $[X]_\mathcal{E}$ ist abgeschlossen gegenüber allen Operationen

3. $[X]_{\mathcal{E}}$ ist die kleinste Menge mit 1. und 2.

. Daraus ergeben sich die drei weiteren Axiome:

1. $0 \in I\!N$,
   in Worten: Null ist eine Zahl.
2. $\forall a(a \in I\!N \rightarrow a' \in I\!N)$,
   in Worten: Jede Zahl besitzt genau einen Nachfolger.
3. $\forall \mathbf{M}(0 \in \mathbf{M} \wedge \forall a(a \in \mathbf{M} \rightarrow a' \in \mathbf{M}) \rightarrow I\!N \subseteq \mathbf{M})$ (Induktionsaxiom)

Satz 4.2 (Fallunterscheidungssatz für Termalgebren)

Ist $\mathcal{E}$ Σ–Termalgebra über X, so gilt für alle $e \in E$: Es gibt ein $s \in S$ und es ist entweder $e \in X_s$ oder es existiert genau ein $\omega \in \Omega$ mit $o_\alpha(\omega) = s$ und genau ein Termtupel $(e_1, e_2, \ldots, e_n) \in E^{i_\alpha(\omega)}$ mit $e = f_\omega(e_1, e_2, \ldots, e_n)$.

Beweis trivial wegen Darstellungssatz für $[X]$ und (P1),(P2).

Folgerung 4.3

Ist $\mathcal{E}$ eine Σ–Termalgebra über X, so gilt für alle $s \in S$:

$$X_s = E_s \setminus \bigcup_{\substack{\omega \in \Omega \\ o_\alpha(\omega) = s}} \operatorname{im} f_\omega$$

Dabei ist $\operatorname{im} f_\omega$ *das volle Bild von f_ω.*

Bemerkung

Mit der Angabe einer Σ–Termalgebra $\mathcal{E} = [E, F]$ ist also zugleich ihr Alphabet X festgelegt.

Definition 4.3

Es sei $\mathcal{A} = [A, F]$ eine Σ–Algebra und $\mathcal{E} = [E, G]$ ein Σ–Ausdrucksalgebra über dem Alphabet X. Jede eindeutige Abbildung(sfamilie) $\varphi : X \mapsto A$ heißt $\mathcal{A}$–*Belegung* von X. Eine eindeutige Abbildung $\varphi^* : E \mapsto A$, die für alle $s \in S$ folgenden Gleichungen

1. $\varphi^*_s(x) = \varphi_s(x)$ für alle $x \in X_s$ und
2. $\varphi^*_s(g_\omega(e_1, e_2, \ldots, e_n)) = f_\omega(\varphi^{*^w}(e_1, e_2, \ldots, e_n))$
   für alle $\omega \in \Omega$ mit $\alpha(\omega) = (w, s)$ und alle $(e_1, e_2, \ldots, e_n) \in E^w$

genügt, heißt *homomorphe* (oder natürliche) *Fortsetzung* der $\mathcal{A}$–Belegung φ von X.

Bemerkung

Falls $\mathcal{E}$ beliebige Σ–Algebra mit dem Erzeugendensystem X wäre, so existierte nicht zu jedem φ eine homomorphe Fortsetzung.

Satz 4.4 (Fortsetzungssatz)

Es seien $\mathcal{A}$ und $\mathcal{E}$ wie oben festgelegt. Dann existiert zu jeder $\mathcal{A}$–Belegung φ des Alphabetes X von $\mathcal{E}$ genau ein Homomorphismus φ^* von $\mathcal{E}$ in $\mathcal{A}$, der über X mit φ übereinstimmt; dieser ist die homomorphe Forsetzung von φ.

Beweis

Die Existenz einer homomorphen Fortsezung φ^* ergibt sich aus dem Fallunterscheidungssatz für Ausdrucksalgebren sofort. Außerdem ist φ^* wegen der zweiten Gleichung Homomorphismus von $\mathcal{E}$ in $\mathcal{A}$. Die Eindeutigkeit folgt aus dem Satz von der Eindeutigkeit der homomorphen Fortsetzung. q.e.d.

$$\begin{tikzcd}
\mathcal{E} \arrow[dr, "\varphi^*"] & \\
 & \mathcal{A} \\
X \arrow[uu, "\subseteq"] \arrow[ur, "\varphi"'] &
\end{tikzcd}$$

Folgerung 4.5

Jede Σ–PEANO–Algebra über X ist also freie Algebra über der Klasse aller Σ–Algebren mit dem freien Erzeugendensystem X.

Satz 4.6 (Isomorphie von Peano–Algebren)

Zwei Σ–PEANO–Algebren $\mathcal{A} = [A, F]$ und $\mathcal{B} = [B, G]$ sind isomorph genau dann, wenn ihre PEANO–Basen X bzw. Y sortenweise gleichmächtig sind.

Beweis

1. Seien X und Y sortenweise gleichmächtig. Nach vorangegangener Folgerung ist $\mathcal{A}$ frei über X, $\mathcal{B}$ frei über Y. Nach dem Satz über die Isomorphie freier Algebren folgt damit $\mathcal{A} \cong \mathcal{B}$.

2. Es sei $\mathcal{A} \cong \mathcal{B}$. Sei weiter $\varphi : A \mapsto B$ der zugehörige Isomorphismus. Wegen der Homomorphieeigenschaft gilt für alle $\omega \in \Omega$: $\varphi(\mathrm{im}\, f_\omega) = \mathrm{im}\, g_\omega$.
Damit ist für alle $s \in S$:

$$\begin{aligned}
\varphi_s(X_s) &= \varphi_s\Big(A_s \setminus \bigcup_{\substack{\omega \in \Omega \\ o_\alpha(\omega) = s}} \mathrm{im}\, f_\omega\Big) \\
&= \varphi_s(A_s) \setminus \bigcup_{\substack{\omega \in \Omega \\ o_\alpha(\omega) = s}} \varphi_s(\mathrm{im}\, f_\omega) \\
&= B_s \setminus \bigcup_{\substack{\omega \in \Omega \\ o_\alpha(\omega) = s}} \mathrm{im}\, g_\omega \\
&= Y_s.
\end{aligned}$$

Da φ_s eineindeutig ist, folgt X_s und Y_s sind gleichmächtig. q.e.d.

4.3 Konstruktion von Termalgebren (synthetische Syntax)

Die folgende Definition liefert eine Konstruktionsmöglichkeit für Ausdrucksalgebren (deswegen die Bezeichnung „synthetische Syntax").

Definition 4.4

Sei $\Sigma = (S, \Omega, \alpha)$ Signatur und sei $X = (X_s)_{s \in S}$ eine disjunkte Mengenfamilie, die auch zu Ω disjunkt ist. Die Mengen $T_{i,s}$, $i \in I\!N$, $s \in S$ seien wie folgt definierte Mengen von endlichen Folgen:

$$\begin{aligned} T_{0,s} &= X_s \cup \{\omega \mid \omega \in \Omega \wedge \alpha(\omega) = (s)\}, \\ T_{i+1,s} &= T_{i,s} \cup \{\omega t_1 t_2 \ldots t_n \mid \omega \in \Omega \wedge \exists w (\alpha(\omega) = (w, s) \wedge (t_1, \ldots, t_n) \in T_i^w)\} \text{ für } i \geq 0. \end{aligned}$$

Dann sei

$$T_s = \bigcup_{i=0}^{\infty} T_{i,s}$$

und

$$T_\Sigma(X) = (T_\Sigma(X)_s)_{s \in S} =_{df} (T_s)_{s \in S}.$$

Bezeichnet f_ω nun für jedes $\omega \in \Omega$ mit $\alpha(\omega) = (w, s)$ die Funktion

$$f_\omega : T_\Sigma(X)^w \to T_\Sigma(X)_s$$

vermöge

$$f_\omega(t_1, t_2, \ldots, t_n) = \omega t_1 t_2 \ldots t_n \text{ für alle } (t_1, \ldots, t_n) \in T_\Sigma(X)^w \text{ bei } w \neq \lambda$$

und

$$f_\omega(\lambda) = \omega \ ,$$

dann ist

$$\underline{T_\Sigma(X)} = [T_\Sigma(X), (f_\omega)_{\omega \in \Omega}]$$

eine Σ–Algebra und heißt *Standardtermalgebra* $\underline{T_\Sigma(X)}$ der Signatur Σ über dem *Variablensystem* (dem Alphabet) X. Eine weitere übliche Bezeichnung ist *Wortalgebra*.

Bemerkung

Wir benutzen hier der Einfachheit halber für die in der Definition vorkommenden Folgen die allgemein übliche „Zeichenreihenschreibweise". Das heißt, die Schreibweise $\omega t_1 \ldots t_n$ ist als Kurzschreibweise der Folge $(\omega, t_1, \ldots, t_n)$ aufzufassen.
Es wird also keinesfalls vorausgesetzt, daß die Elemente von X oder von Ω in irgendeinem Sinne „Zeichen" eines Alphabets seien. Wer dies allerdings wünscht, kann auch mit dieser Auffassung leben.

Bemerkung

Da ω die oben definierte Operation f_ω eindeutig festlegt, wird diese in der Literatur oft selbst mit ω bezeichnet, und damit die Standardtermalgebra durch

$$\underline{T_\Sigma(X)} = [T_\Sigma(X), \Omega]$$

.

Satz 4.7

Die Standardtermalgebra $\underline{T_\Sigma(X)}$ über X ist eine PEANO*–Algebra über X.*

Beweis

Der Beweis ist sehr einfach: per definitionem $X \subseteq T_\Sigma(X)$. Es bleiben die verallgemeinerten Peano-Axiome nachzuweisen.

P1: $f_\omega(t_1, \ldots, t_n) = \omega t_1 \ldots t_n$, das ist also stets eine Folge, deren erste Komponente ω ist. Die Disjunktheitsforderung von X und Ω garantiert also in jedem Fall, daß $\omega t_1 \ldots t_n \notin X$.

P2: Gilt $f_\omega(t_1, \ldots, t_n) = f_\psi(t'_1, \ldots, t'_m)$ für $\omega, \psi \in \Omega$, so sind diese beiden Folgen $\omega t_1 \ldots t_n$ und $\psi t'_1 \ldots t'_m$ gleich. Nach dem bekannten Hauptsatz über Folgen ist also $\omega = \psi$ und $m = n$ und $t_i = t'_i$ für $1 \leq i \leq n$.

P3: $[X]_{\underline{T_\Sigma(X)}} = T_\Sigma(X)$ gilt entsprechend der Definiton von $T_\Sigma(X)$ auf Grund des Darstellungsatzes für die algebraischen Hülle.

q.e.d.

Folgerung 4.8

*Jede Σ–*PEANO*–Algebra über X ist isomorph zur Standardtermalgebra $\underline{T_\Sigma(X)}$*

Beweis

$\underline{T_\Sigma(X)}$ ist PEANO–Algebra über X. Zwei PEANO–Algebren über derselben Basis sind aber nach dem Isomorphiesatz für PEANO–Algebren trivialerweise isomorph. q.e.d.

Bemerkung

In der Definition von $\underline{T_\Sigma(X)}$ kann X auch als S-sortige Familie leerer Mengen $X = \underline{\emptyset} = (\emptyset)_{s \in S}$ gewählt werden.

Wir setzen:

Definition 4.5

$$T_\Sigma =_{df} T_\Sigma(\underline{\emptyset}) \quad \text{und} \quad \underline{T_\Sigma} =_{df} \underline{T_\Sigma(\underline{\emptyset})}.$$

Damit wird T_Σ nur mit Hilfe der Operatoren, ausgehend von den Konstanten, aufgebaut.

Der Fortsetzungssatz für Ausdrucksalgebren wird nun in diesem Spezialfall zu folgendem

Satz 4.9

Für jede Σ–Algebra $\mathcal{A}$ gibt es genau einen Homomorphismus

$$h_{\mathcal{A}} : \underline{T_\Sigma} \mapsto \mathcal{A}.$$

Bemerkung

Eine solche Algebra nennt man in der Kategorientheorie bzw. in der Universellen Algebra eine *initiale* Algebra in der entsprechenden Algebrenklasse, d. h. $\underline{T_\Sigma}$ ist initiale Algebra in der Klasse aller Σ–Algebren.

Lemma 4.10

Ist die Σ–Algebra $\mathcal{A} = [A, F]$ über der Klasse aller Σ–Algebren frei mit dem freien Erzeugendensystem X, und X disjunkt zu Ω, und möge φ der eindeutig bestimmte Homomorphismus von $\mathcal{A}$ in $\underline{T_\Sigma(X)}$ sein mit $\varphi\mid_X = \iota_X$, so ist φ injektiv (also umkehrbar eindeutig).

Beweis

(durch algebraische Induktion)
Sei $M = (M_s)_{s\in S}$ und

$$M_s = \{t \mid t \in T_s \wedge \forall \mathsf{a} \forall \mathsf{a}'(\varphi(\mathsf{a}) = \varphi(\mathsf{a}') = t \rightarrow \mathsf{a} = \mathsf{a}'\}$$

1. $X_s \subseteq M_s$ denn sei $\varphi(\mathsf{a}) = \varphi(\mathsf{a}') = x$, $x \in X_s$, so ist $\mathsf{a} \notin X$ oder $\mathsf{a}' \notin X$ ausgeschlossen. Wäre nämlich z. B. $\mathsf{a} \notin X$, so existiert ein $\omega \in \Omega$, $w \in S^*$ mit $\mathsf{a} = f_\omega(a_1, \ldots, a_n)$. (d.h. wenn es nicht aus dem Erzeugendensystem ist, so muß es erzeugt sein.) Daraus folgt $\varphi(\mathsf{a}) = \varphi(f_\omega(a_1, \ldots, a_n)) = \omega\varphi(\mathsf{a}_1)\ldots\varphi(\mathsf{a}_n) = x$. Dies ist aber eine Widerspruch zur Folgengleichheit, Disjunktheit. Also folgt $\mathsf{a} \in X$ und $\mathsf{a}' \in X$. Über X ist φ injektiv nach Voraussetzung.
2. Sei $\omega \in \Omega$, $\alpha(\omega) = (w, s)$ und $(t_1, \ldots, t_n) \in M^w$. Wir betrachten $t = \omega t_1 \ldots t_n$. Angenommen, es existieren $a, a' \in A$ mit $\varphi(a) = \varphi(a') = t = \omega t_1 \ldots t_n$. Dann existieren wegen der Homomorphieeigenschaft $(a_1, \ldots, a_n) \in A^w$ und $(a'_1, \ldots, a'_n) \in A^w$ mit $a = f_\omega(a_1, \ldots, a_n)$ und $a' = f_\omega(a'_1, \ldots, a'_n)$ und $t = \omega t_1 \ldots t_n = \varphi(a) = \omega\varphi(a_1)\ldots\varphi(a_n) = \varphi(a') = \omega\varphi(a'_1)\ldots\varphi(a'_n)$, d. h. wegen der Folgengleichheit $t_i = \varphi(a_i) = \varphi(a'_i)$ für $1 \leq i \leq n$, woraus wegen $t_i \in M$ also $a_i = a'_i$ folgt. Damit ist auch $a = a'$, d. h. $t \in M_s$.

Nach dem Prinzip der algebraischen Induktion ist $M = T_\Sigma(X)$, d.h. φ ist injektiv. q.e.d.

Satz 4.11

*Ist die Σ–Algebra $\mathcal{A} = [A, F]$ über der Klasse aller Σ–Algebren frei mit dem zu Ω disjunkten freien Erzeugendensystem X, so ist $\mathcal{A}$ eine Σ–*PEANO*–Algebra über der* PEANO*–Basis X.*

Beweis

1. X ist nach Voraussetzung Erzeugendensystem von $\mathcal{A}$, also ist **P3** erfüllt.
2. Nach Punkt 1 im Beweis des obigen Lemmas folgt für beliebiges $\omega \in \Omega$ mit $o_\alpha(\omega) = s$, daß im $f_\omega \cap X_s = \emptyset$. Damit ist auch **P1** erfüllt.
3. Angenommen, für ein $s \in S$ existieren $\omega_1, \omega_2 \in \Omega$ mit $\alpha(\omega_1) = (w_1, s)$ und $\alpha(\omega_2) = (w_2, s)$ und es existieren $\mathsf{a} \in A^{w_1}$, $\mathsf{b} \in A^{w_2}$, so daß $f_{\omega_1}(\mathsf{a}) = f_{\omega_2}(\mathsf{b})$. Wir betrachten $\underline{T_\Sigma(X)}$ und den nach Voraussetzung (Freiheit von $\mathcal{A}$) zu $\iota_X : X \rightarrow T_\Sigma(X)$ eindeutig existierenden Homomorphismus φ : $\mathcal{A} \mapsto \underline{T_\Sigma(X)}$ mit $\varphi\mid_X = \iota_X$. Dann gilt $\varphi(f_{\omega_1}(\mathsf{a}) = \varphi(f_{\omega_2}(\mathsf{b}))$, d.h. $\omega_1\varphi^{w_1}(\mathsf{a}) = \omega_2\varphi^{w_2}(\mathsf{b})$, woraus wegen der Folgengleichheit $\omega_1 = \omega_2$ und $\varphi^{w_1}(\mathsf{a}) = \varphi^{w_2}(\mathsf{b})$ folgt. Nach obigen Lemma (φ injektiv) folgt auch $\mathsf{a} = \mathsf{b}$, also ist **P2** erfüllt.

q.e.d.

Folgerung 4.12

1. *Ist $\mathcal{A}$ eine freie Algebra mit einem freien Erzeugendensystem X, so ist dieses eindeutig bestimmt (wegen der eindeutigen Bestimmtheit der* PEANO*–Basis bei* PEANO*–Algebren).*
2. *Der Satz über die Isomorphie freier Algebren läßt sich verschärfen:*
   Freie Algebren $\mathcal{A}$ und $\mathcal{B}$ sind isomorph genau dann, wenn *ihre freien Erzeugendensysteme sortenweise gleichmächtig sind.*

4.4 Drei Charakterisierungen

Zusammenfassend ergibt sich, daß die drei Charakterisierungsarten für Algebren, die eine homomorphe Fortsetzung beliebiger „Belegungen" eines Erzeugendensystems erlauben, äquivalent sind:

Satz 4.13 (über die Termalgebren)

Für jede Σ-Algebra $\mathcal{A} = [A, F]$ und beliebige Teilmengen $X \subseteq A$ sind folgende Aussagen äquivalent:

1. *$\mathcal{A}$ ist Σ-*PEANO*-Algebra über der* PEANO*-Basis X.*
2. *$\mathcal{A}$ ist frei in der Klasse aller Σ-Algebren mit dem freien Erzeugendensystem X.*
3. *$\mathcal{A}$ ist isomorph zu $\underline{T_\Sigma(X)}$ vermöge eines X festlassenden Isomorphismus.*

Charakterisierung	
innere	*analytisch*
äußere	*algebraisch*
konstruktive	*synthetisch*

Beweis

Schon gezeigt wurden folgende Implikationen: (1) $\rightarrow$ (2) (Folgerung zum Fortsetzungssatz), (2) $\rightarrow$ (1) (letzter Satz), (1) $\rightarrow$ (3) (Folgerung zu: $\underline{T_\Sigma(X)}$ ist PEANO-Algebra).

Um die paarweise Äquivalenz der Aussagen zu erhalten, zeigen wir jetzt noch: (3) $\rightarrow$ (2).

Sei $\psi : \mathcal{A} \mapsto \underline{T_\Sigma(X)}$ ein Isomorphismus mit $\psi\mid_X = \iota_X$. Weiter sei $\mathcal{B}$ eine beliebige Σ-Algebra und $\varphi : X \mapsto B$ eine beliebige $\mathcal{B}$-Belegung von X. Weil $\underline{T_\Sigma(X)}$ PEANO-Algebra ist, existiert nach dem Fortsetzungssatz genau ein Homomorphismus

$$\varphi^* : \underline{T_\Sigma(X)} \mapsto \mathcal{B} \text{ mit } \varphi^*\mid_{\mathcal{X}} = \varphi.$$

$\sigma = \varphi^* \circ \psi$ ist damit als Nacheinanderausführung von Homomorphismen wieder ein Homomorphismus, und zwar von $\mathcal{A}$ in $\mathcal{B}$. Dabei ist

$$\sigma\mid_X = (\varphi^* \circ \psi)\mid_X = \varphi^* \circ \psi\mid_X = \varphi^* \circ \iota_X = \varphi^*\mid_X = \varphi,$$

d.h. σ ist homomorphe Fortsetzung von φ. Also ist $\mathcal{A}$ frei über X. q.e.d.

4.5 Beispiele für Ausdrucksalgebren

1. Algebra der aussagenlogischen Ausdrücke:

   $$[\underline{AUS}\,;\, neg\,,\, con\,,\, alt]$$

   (vgl. vorn die Beispiele für Σ–Algebren) ist ein klassisches Beispiel für eine homogene Ausdrucksalgebra.

2. PEANO-Algebra der natürlichen Zahlen:
   Sie ist eine weitere homogene Ausdrucksalgebra. Zur Begründung können die Bemerkungen zu den verallgemeinerten PEANO–Axiomen herangezogen werden. Wir betrachten den Algebrentyp τ mit

   $$\Omega = \{O\,,\,'\}, \quad \tau(O) = 0,\, \tau(') = 1,$$

   bezeichnen aber die sich ergebende Signatur wie üblich mit Σ.

   (a) Terme der Standardtermalgebra $\underline{T_\Sigma}$ wären dann z.B.:

   $$O,\ 'O,\ ''O,\ \ldots$$

   (b) Isomorph dazu und damit auch initial in der Klasse aller Σ–Algebren ist natürlich die Termalgebra mit den Termen:

   $$O,\ O',\ O'',\ \ldots$$

   (c) Ausdrucksalgebra derselben Signatur über der Variablenmenge $\{x_1, x_2, \ldots\}$
   mit den Ausdrücken:

   $$x_1,\ x_2,\ x_{27},\ O,\ O'''',\ x_{51}''''' \quad \text{usw.}$$

   (d) Selbstverständlich ist die Algebra $\mathcal{N}$ vom Typ $[0,1]$:

   $$\mathcal{N} = [I\!N;\ O,\ \sigma],$$

   wobei $I\!N$ die Menge der natürlichen Zahlen, O die Zahl 0 und σ die Nachfolgerfunktion bezeichnet, zu den unter 2a bzw. 2b angedeuteten Algebren isomorph und ist damit ebenfalls Ausdrucksalgebra sowie ebenfalls initial in der Klasse aller homogenen Algebren vom Typ $[0,1]$.
   Dies ist unabhängig von der konkreten Gestalt der natürlichen Zahlen, d.h. unabhängig davon, ob

   $$I\!N = \{0, 1, 2, \ldots\},$$

   $$I\!N = \{\mathsf{I}, \mathsf{II}, \mathsf{III}, \mathsf{IIII}, \ldots\},$$

   $$I\!N =_{df} \textit{ Menge der Äquivalenzklassen gleichmächtiger endlicher Mengen},$$

   $$I\!N = O, L, LO, LL, LOO, \ldots,$$

   $$I\!N =_{df} \textit{ Menge der Ordinalzahlen},$$

   oder was auch immer man unter den „natürlichen Zahlen“ versteht.

3. Die im folgenden erklärte Algebra einfacher arithmetischer Terme ist eine weitere homogene Ausdrucksalgebra:
   Dazu seien

   $$X = \{x_1, x_2, \ldots\}, \quad \Omega = \{0, +\} \qquad \text{und das Hilfsalphabet } H = \{(,)\}$$

   Alphabete im Sinne der Theorie der Zeichenreihen. Weiter sei

   $$\tau = [0, 2], \qquad \text{also } \tau(0) = 0, \qquad \tau(+) = 2.$$

   Betrachte nun die Algebra

   $$\mathcal{E} = [E\ ;\ f_0, f_+]$$

(a) $f_0 = 0, \quad f_+(e_1, e_2) \;=\; (e_1 + e_2)$ für beliebige $e_1, e_2 \in E$.
Dann sind Ausdrücke (Elemente von E) beispielsweise:

$$0,\ ((x_1 + 0) + x_3),\ (x_1 + (0 + x_2)),\ (x_2 + x_1)\ldots$$

(b) H weglassen, f_+ mittels Präfixschreibweise erklären: $f_+(e_1, e_2) \;=\; +e_1 e_2$
Ausdrücke sind dann:

$$0,\ + + x_1 0 x_3,\ +x_1 + 0x_2\ \ + x_2 x_1$$

4. Eine homogene Termalgebra über $X = \{p, q, r\}$ mit $\Omega = \wedge, \vee$ vom Typ $[2, 2]$:
Terme sind zum Beispiel:

(a) unter Benutzung der üblichen Folgenschreibweise (vgl. die Definition der Standardtermalgebra: die dortigen Terme sind *Folgen*):

$$(p),\ (\wedge, (\wedge, q, r), p),\ (\wedge, (\vee, p, r), (\wedge, p, p))$$

(b) unter Verwendung von Zeichenreihenschreibweisen:

Präfixschreibweise:	p,	$\wedge \wedge q\, r\, p$,	$\wedge \vee p\, r \wedge p\, p$
Infixschreibweise:	p,	$((q \wedge r) \wedge p)$,	$((p \vee r) \wedge (p \wedge p))$
Postfixschreibweise:	p,	$q\, r \wedge p \wedge$,	$p\, r \vee p\, p \wedge \wedge$
Funktionsschreibweise:	p,	$\wedge(\wedge(q, r), p)$,	$\wedge(\vee(p, r), \wedge(p, p))$

5. Zum Mealy-Automaten (vgl. vorn Beispiel 1):

$$\Sigma = (S, \Omega, \alpha), \quad S = \{x, y, z\}, \quad \Omega = \{1, 2\},$$

$$\alpha(1) = (z, x, z), \quad \alpha(2) = (z, x, y).$$

Bilde Σ-Standardtermalgebra über $V = (V_s)_{s \in S}$ mit $V_x = \{x_1, x_2, \ldots\}$, $V_y = \{y_1, y_2, \ldots\}$ als abzählbar unendliche Alphabete und $V_z = \{z_1, z_2, z_3\}$ als endliches Alphabet.

Zu $T_\Sigma(V)$ gehören z. B. folgende Terme:

T_x: Terme der Sorte x („Eingabeterme“):	$x_1, x_2, x_3, \ldots$	(nur das „Eingabealphabet“)
T_y: Terme der Sorte y („Ausgabeterme“):	$y_1, y_2, y_3, \ldots$	
	$2z_1x_1,\ 2z_1x_2,\ 2z_1x_3, \ldots$	
	$2z_2x_1,\ 2z_2x_2,\ 2z_2x_3, \ldots$	
	$2z_3x_1,\ 2z_3x_2,\ 2z_3x_3, \ldots$	
	$2\,1z_1x_1x_1,\ 2\,1z_1x_1x_2, \ldots$	
	$2\,\underbrace{1z_1x_2}_{\in T_z}x_1,\ 2\,1z_1x_2x_2, \ldots$	
	$\vdots$	
T_z: Terme der Sorte z („Zustandsterme“):	$z_1, z_2, z_3, \ldots$	
	$1z_1x_1,\ 1z_1x_2,\ 1z_1x_3, \ldots$	
	$1z_2x_1,\ 1z_2x_2,\ 1z_2x_3, \ldots$	
	$1z_3x_1,\ 1z_3x_2,\ 1z_3x_3, \ldots$	
	$1\,1z_1x_1x_1,\ 1\,1z_1x_1x_2, \ldots$	
	$\vdots$	

6. Zum Datentyp Keller:

$$\Sigma = (S, \Omega, \alpha) \ \text{ mit } S = \{d, s\} \quad (d \text{ stehe für } \textit{date}, \text{ s für } \textit{stack}).$$

Weiter sei

$$\Omega = \{*, \#, \Lambda, \mathsf{pop}, \mathsf{push}, \mathsf{top}\}$$

mit $*$ als *date error*, $\#$ als *stack error* und Λ als *leerer Keller*.

$$\alpha(*) = (d),\ \alpha(\#) = \alpha(\Lambda) = (s),$$

$$\alpha(\mathsf{push}) = (d, s, s),\ \alpha(\mathsf{pop}) = (s, s),\ \alpha(\mathsf{top}) = (s, d).$$

Bilde in leichter Abwandlung der Definition der Standardtermalgebra eine Ausdrucksalgebra $E_\Sigma(X)$ über $X = \{X_d, X_s\}$ mit

$$X_d = \{x_1, x_2, \ldots\},\ X_s = \emptyset$$

Dann gehören zu

E_s zum Beispiel: $\#,\ \Lambda,\ \mathsf{push}(x_i \mathsf{push}(x_j \Lambda)),\ \mathsf{pop}(\mathsf{push}(x_i \Lambda))$

E_d zum Beispiel: $x_1, \ldots,\ *,\ \mathsf{top}(\Lambda),\ \mathsf{top}(\mathsf{push}(x_1 \Lambda)),\ \ldots$

7. Ein Beispiel von selbständiger Bedeutung:
<u>Zuordnung der syntaktischen Algebra zu einer gegebenen kontextfreien Grammatik:</u>

$G = (N, T, P, S)$ sei eine beliebige kontextfreie Grammatik.

Dabei bedeuten

N	–	Alphabet der Nichtterminale,
T	–	Alphabet der Terminale
P	–	Menge der Produktionen bzw. Regeln: $P \subseteq N \times (T \cup N)^*$
S	–	Startsymbol: $S \in N$.

Sei für jedes $A \in N$

$$L_A = \{w \mid A \xrightarrow{*} w \wedge w \in T^*\}$$

(Anmerkung: $L(G) = L_S$)

Für jede Regel $p \in P$ der folgenden Gestalt

$$A \longrightarrow u_1 A_1 u_2 A_2 \ldots u_n A_n u_{n+1} \text{ mit } A, A_i \in N,\ u_k \in T^*$$

sei eine Operation f_p wie folgt definiert:

$$f_p : L_{A_1} \times \cdots \times L_{A_N} \mapsto L_A$$

mit $f_p(x_1, \ldots, x_n) = u_1 x_1 u_2 x_2 \ldots u_n x_n u_{n+1}$.

Wir bezeichnen nun

$$\underline{\mathsf{SYN}}_G = [(L_A)_{A \in N}; (f_p)_{p \in P}]$$

als die zu G gehörige *syntaktische Algebra*. Vergleiche dazu auch das Einführungsbeispiel (Abschnitt 1.2). Die Signatur von $\underline{\mathsf{SYN}}_G$ ist

$$\Sigma_G = (N, P, \alpha)$$

mit $\alpha(p) = (r(p)^T/_\varepsilon, l(p))$, wobei $r(p), l(p)$ die rechte bzw. linke Seite der Regel p bezeichnet und $T/_\varepsilon$ die Streichung aller Terminale aus $r(p)$ andeuten soll.

Sind nun alle Trägersprachen L_A (nicht nur $L(G)$ selbst) von $\underline{\mathsf{SYN}}_G$ im Sinne der Theorie der formalen Sprachen *eindeutige* Sprachen, dann erfüllt $\underline{\mathsf{SYN}}_G$ die verallgemeinerten PEANO-Axiome P1, P2, P3 und ist damit eine PEANO- bzw. Ausdrucksalgebra.
Das eindeutig bestimmte Erzeugendensystem X ist dabei das System der leeren Mengen $\emptyset$.
Beachte:

$$X_A = L_A \setminus \bigcup_{l(p)=A} \mathrm{im}(f_p) = \emptyset$$

Die Signatur Σ_G, die durch eine kontextfreie Grammatik G festgelegt ist, bestimmt wiederum die initiale Standardtermalgebra $\underline{T_{\Sigma_G}}$, kurz $\underline{T_G}$.
Deren Trägermengen $T_{G,A}$ bestehen aus allen Folgen von Regeln, die zu vollständigen Linksableitungen in G, beginnend mit A, gehören. Da nun - wie aus der Theorie der formalen Grammatiken bekannt - jeder Linksableitung eineindeutig ein Ableitungsbaum entspricht, kann man also sagen, daß die initiale Termalgebra der Signatur Σ_G gerade die *Ableitungsbäume* als ihre Elemente besitzt.
Deshalb wird diese Algebra auch *abstrakte Syntax* genannt.

Wir wollen diesen Sachverhalt an einem kleinen *Beispiel* verdeutlichen:
Sei $\Sigma_G \;=\; (\{A, B\},\, P,\, \alpha)$, wobei zu P u. a. gehöre:

„Marke“	Regel	Arität α	
$+$:	$A \longrightarrow (A+B)$	$(A\,B,\ A)$	
a :	$A \longrightarrow a$	(A)	Konstante!
b :	$B \longrightarrow b$	(B)	-“-
$-_1$:	$A \longrightarrow -A$	$(A,\ A)$	
$-_2$:	$A \longrightarrow (A-A)$	$(A\,A,\ A)$	

Dann gehört zu $T_{G,B}$ nur die „Konstante“: $B \longrightarrow b$
und zu $T_{G,A}$ z.B.:

$$A \longrightarrow a$$

$$\underbrace{A \longrightarrow (A+B)}_{\omega}\quad \underbrace{A \longrightarrow a}_{t_1}\quad \underbrace{B \longrightarrow b}_{t_2}$$

oder

$$\underbrace{A \longrightarrow (A+B)}_{\omega}\quad \underbrace{A \longrightarrow (A+B)\quad A \longrightarrow a\quad B \longrightarrow b}_{t_1}\underbrace{B \longrightarrow b}_{t_2}$$

oder auch

$$\underbrace{A \longrightarrow -A}_{\omega}\quad \underbrace{\underbrace{A \longrightarrow (A+B)}_{\omega_2}\quad \underbrace{\underbrace{A \longrightarrow (A-A)}_{\omega_3}\quad \underbrace{A \longrightarrow a}_{t_{31}}\quad \underbrace{A \longrightarrow a}_{t_{32}}}_{t_{21}}\quad \underbrace{B \longrightarrow b}_{t_{22}}}_{t}$$

Deutlicher wird der Zusammenhang mit den Ableitungsbäumen, wenn wir statt der Regeln als Operatoren deren oben angeführte „Benennungen“, d. h. die „Marken“ verwenden. Dieselben Terme von $T_{G,A}$ wie oben sehen dann wie folgt aus:

a
$+a\,b$
$+ + a\,b\,b$
$-_1 + -_2 a\,a\,b$

Daß diese Standardterme in eindeutiger Weise bestimmten Ableitungsbäumen entsprechen, ist wohl offensichtlich.

Kapitel 5

Syntax und Semantik von Sprachen

5.1 Zur Syntax

Die Beispiele zu Ausdrucksalgebren, besonders die Beispiele 1, 3, 4 und 7 (kontextfreie Sprachen), haben gezeigt, daß die Syntax formaler Sprachen aufgefaßt bzw. gegeben werden kann als eine i. a. mehrsortige Algebra. Die Trägermengen sind dabei die Mengen unterschiedlicher syntaktischer Objekte, die Operationen diejenigen syntaktischen Operationen, die aus bereits konstruierten Teilobjekten neue syntaktische Objekte machen. Die entsprechende Algebra wird auch *syntaktische Algebra* genannt.
Als Programmierer, Informatiker, Logiker usw. ist man schnell daran gewöhnt, bei vielen Überlegungen die Ebenen der Syntax und der Semantik wohl zu unterscheiden. Aber schon auf der Ebene der Syntax hat man genaugenommen zwei Niveaustufen zu unterscheiden:

- die Ebene der „*Sprachstruktur*“, der eigentlichen Grammatik der Sprache,
- die Ebene der „*Rechtschreibung*“ oder Orthographie.

Die Existenz der beiden syntaktischen Ebenen und ihre Unterschiedlichkeit wird oft übersehen bzw. auch nicht erkannt. Dies zeigt auch der Umstand, daß die zweite Ebene unter Sprachentwerfern oft als „syntaktischer Zucker“ bezeichnet wurde. Viele halten ausschließlich die Semantik (einer Programmiersprache) für das wesentliche und beschäftigen sich nur deshalb mit der Syntax, weil die Semantik ohne diese nicht zu beschreiben ist.
Ein kleines **Beispiel** mag das Wesen der beiden Ebenen erläutern:
Angenommen, man habe eine Programmiersprache zu entwerfen, die natürlich auch bedingte Anweisungen enthalten soll. Zuerst hat man sich da zu entscheiden, welche Varianten man zulassen will, z. B.:

- ein (Boolescher) Ausdruck, eine Anweisung
  (entspricht den bekannten “unvollständigen bedingten Anweisungen“)

- ein Ausdruck, zwei Anweisungen
  (Ja–Nein–Verzweigung)

- eine beliebige Zahl von Paaren Ausdruck – Anweisung
  (etwa als n–zweigige Fallunterscheidung oder im Sinne der
  sogenannten Dijkstraschen *guarded commands*)

Das ist die rein „grammatikalische“ bzw. *strukturelle Ebene.*

Entscheidet man sich nun z. B. für die zweite Variante, so könnte man beispielsweise unter folgenden Notationsformen wählen:

- Cond ::= **if** Exp **then** Com **else** Com **fi**
- Cond ::= **case** Exp **true** Com **false** Com
- Cond ::= Exp $\longrightarrow$ Com , Com
- Cond ::= **(** Exp | Com | Com **)**

Das ist dann die *Rechtschreibungsebene.*

Man beachte, daß die üblicherweise zur Definition der Syntax von Programmiersprachen gebrauchte Backus–Naur–Form nicht erlaubt, die erste Ebene der Sprachstruktur unabhängig von der zweiten Ebene der Orthographie zu definieren!

Die strukturelle Ebene, die auch *abstrakte Syntax* genannt wird, (vgl. Beispiel 7) „trägt" die Bedeutung, die Rechtschreibung ist mehr pragmatischer Natur.
Algebraisch ist die *abstrakte Syntax* die (in irgendeiner Weise ausgezeichnete) *initiale Standardtermalgebra* der durch die syntaktische Algebra festgelegten Signatur bzw. (will man keine besonders auszeichnen) die *Isomorphieklasse der initialen Algebren* der entsprechenden Signatur. Da die Angabe der Isomorphieklasse in aller Regel durch einen ausgezeichneten Repräsentanten erfolgen wird, kann man dann auch gleich diesen dann mit der abstrakten Syntax identifizieren.

5.2 Zur Semantik

Entsprechend den Sätzen über die Ausdrucksalgebren existiert zu vorgegebener Algebra gleicher Signatur genau ein Homomorphismus von der abstrakten Syntax in die gegebene Algebra. Das ist der Kerngedanke der sogenannten „Algebraischen Semantik", die manchmal auch „Mathematische Semantik" genannt wird. Jede Algebra derselben Signatur wie die syntaktische Algebra (und die abstrakte Syntax) kann als *semantische Algebra* fungieren. Der dadurch eindeutig festgelegte Homomorphismus heißt *semantischer Homomorphismus* und vermittelt eben die *Semantik* der Spache, d. h. er ordnet den syntaktischen Objekten Bedeutungen zu. Als Homomorphismus verwirklicht er dabei das sogenannte

Fregesche Prinzip:

„Die Bedeutung des Ganzen ist eine Funktion der Bedeutung seiner Teile".

Man beachte, daß bei Eindeutigkeit der Sprache die syntaktische Algebra $\underline{\mathsf{SYN}}$ ebenfalls eine Ausdrucksalgebra und damit isomorph zur abstrakten Syntax $\underline{\mathrm{ABS}}$ ist. Im allgemeinen ist (grammatikalische) Eindeutigkeit bei Programmiersprachen unerläßlich. Folglich ist die Festlegung der Rechtschreibung nicht vollkommen willkürlich, sie hat neben pragmatischen Aspekten insbesondere die Eindeutigkeit der Sprache zu sichern. Dazu dient speziell die geeignete Verwendung von Klammern, Trennzeichen usw.

$$\begin{tikzcd} \underline{\mathsf{SYN}} \arrow[rr, "\psi"] & & \underline{\mathsf{SEM}} \\ & \underline{\mathrm{ABS}} \arrow[ul, "\varphi"] \arrow[ur, "\sigma = \text{semantischer Homomorphismus}"'] & \end{tikzcd}$$

$$\psi = \sigma \circ \varphi^{-1}$$

φ^{-1} ist bei Eindeutigkeit der Sprache auch Homomorphismus!

Bei sogenannter „semantischer Eindeutigkeit" sind Ausnahmen von der Forderung der grammatikalischen Eindeutigkeit möglich, darauf soll jetzt noch kurz eingegangen werden.
Allgemein ist bei Mehrdeutigkeit der Sprache $\underline{\mathrm{SYN}}$ nicht mehr Peano–Algebra, folglich ist φ i. a. nicht mehr umkehrbar eindeutig. Will man unter diesen Umständen die Bedeutung eines (sprachlichen) Elementes $e \in \underline{\mathsf{SYN}}$ ermitteln, so kann es (mindestens) zwei Elemente $a_1, a_2 \in \underline{\mathsf{ABS}}$ geben (dies entspricht zwei

verschiedenen grammatikalischen Analysen bzw. zwei Ableitungsbäumen), für die gilt

$$\varphi(a_1) = \varphi(a_2) = e.$$

Dann wird es im allgemeinen zwei „Bedeutungen", das heißt, zwei Elemente $b_1, b_2 \in \underline{\mathsf{SEM}}$ geben, für die

$$\sigma(a_1) = b_1 \text{ und } \sigma(a_2) = b_2$$

ist. Damit könnte man sowohl

$$\psi(e) = b_1 \text{ als auch } \psi(e) = b_2$$

setzen, eine für Programmiersprachen generell untragbare Situation. Nur in dem Fall, daß in der semantischen Algebra $\underline{\mathsf{SEM}}$ gilt

$$b_1 \;=\; b_2 \qquad\qquad \textit{semantische Eindeutigkeit}\;,$$

wäre dies zulässig.
Das wird in der Praxis tatsächlich ausgenutzt, man denke beispielsweise an die Verkettung von Anweisungen mittels Semikolon, die assoziativ verwendet wird. Das heißt, im allgemeinen muß es bei der Anweisungsfolge

$$A_1 \;;\; A_2 \;; \ldots ; A_n$$

nicht interessieren, ob eine Anweisung A_1 mit einer Anweisungsfolge $A_2 \;; \ldots ; A_n$ oder eine Anweisungsfolge $A_1 \;;\; A_2 \;; \ldots ; A_{n-1}$ mit einer Anweisung A_n oder gar zwei Anweisungsfolgen $A_1 \;;\; A_2 \;; \ldots ; A_k$ und $A_{k+1} \;; \ldots ; A_n$ miteinander verkettet wurden, eben weil die Semantik der Verkettung ebenfalls *assoziativ* ist! Ein typischer Fall von sogenannter semantischer Eindeutigkeit.

Fortsetzung des Beispiels 7 (kontextfreie Grammatik)

$\underline{T_G}$ ist also die abstrakte Syntax der durch die Grammatik G gegebenen Signatur Σ_G, sie ist - wie erläutert - die Algebra der Ableitungsbäume.

Wir konstruieren nun eine weitere Σ_G–Algebra, zu der dann ein eindeutig bestimmter Homomorphismus von T_G in diese existieren muß.
Dazu wählen wir als Trägermengen (alle gleich) die Wortmenge T^* über dem Terminalalphabet T. Als Operationen fungieren die durch die Produktionen p gegebenen weiter oben beschriebenen Funktionen f_p, nun aber ausgedehnt auf T^*:

$$f_p: \quad T^* \times \ldots \times T^* \;\mapsto\; T^*$$

$$\text{mit} \quad f_p(w_1, \ldots, w_n) \;=\; u_1 w_1 u_2 \ldots u_n w_n u_{n+1}$$

$$\text{bei} \quad p: \quad A \longrightarrow u_1 A_1 u_2 A_2 \ldots u_n A_n u_{n+1}.$$

Die erhaltene Σ_G–Algera sei durch $\underline{D_G}$ bezeichnet, es ist also

$$\underline{D_G} \;=\; [\,(T^*)_{A \in N}\,;\,(f_p)_{p \in P}\,].$$

Dann ordnet der eindeutig bestimmte Homomorphismus $h_D \;:\; \underline{T_G} \;\mapsto\; \underline{D_G}$ jedem „Ableitungsbaum" der Sorte A_i (d. h. mit Wurzel A_i) die durch die entsprechende Linksableitung erzeugte Zeichenreihe aus T^*, d. h. das abgeleitete Wort aus L_{A_i} zu!

Wir verdeutlichen das an dem schon oben verwendeten kleinen *Beispiel*:
Betrachte $+ + a\,b\,a \quad \in \; T_{G,A}$:

$$\begin{aligned}
h_{D,A}(+ + a\,b\,a) \;&=\; f_+(h_{D,A}(+a\,b)\,, h_{D,A}(a)\,)\\
&=\; f_+(\,f_+(\,h_{D,A}(a)\,,\,h_{D,A}(b)\,)\,,\,f_a)\\
&=\; f_+(\,f_+(\,f_a\,,\,f_b)\,,\,f_a)\\
&=\; f_+(f_+(a,b),a)\\
&=\; f_+(\,(a+b)\,,\,a)\\
&=\; ((a+b)+a)
\end{aligned}$$

Als homomorphes Bild $h_D(\underline{T_G})$ erhalten wir hier also die Ausgangsalgebra $\underline{\mathsf{SYN}}_G$ und erkennen gleichzeitig, daß sie eine Unteralgebra von $\underline{D_G}$ ist.
Mit dem oben eingeführten Sprachgebrauch stellt also die „Zeichenreihenerzeugung" selbst eine *„Semantik"*

der durch die Grammatik G festgelegten Syntax $\underline{T_G}$ dar!
Dies zeigt nur einmal wieder, daß es sehr von der Betrachtungsweise abhängt, was man als zum syntaktischen Bereich, was als zum semantischen Bereich gehörig anzusehen hat.

5.3 Die „Algebraische Semantik"

Sicherlich kann man - wie oben - als eine Bedeutung einer Grammatik (bzw. der durch sie festgelegten abstrakten Syntax) die Erzeugung der Sprachen ansehen. Mehr aber wird die „Bedeutung" dieser erzeugten Sprachen interessieren. Diese kann man als eine Semantik wieder der abstrakten Syntax oder auch (vermöge des entsprechenden Isomorphismus) der erzeugten Sprachen, d. h. der syntaktischen Algebra, ansehen.
Entsprechend dem Fortsetzungssatz für Ausdrucksalgebren kann **jede** Algebra der betrachteten Signatur als *semantische Algebra* $\underline{\mathsf{SEM}}$ fungieren.
Anliegen der sogenannten „Algebraischen Semantik" (in der englischsprachigen Literatur als "algebraic semantics" bezeichnet) und auch der „denotationalen Semantik" ist es nun, die jeweils *beabsichtigte* Semantik durch geeignete Beschreibung („Spezifikation") der gemeinten semantischen Algebra zu definieren. Der dann feststehende Homomorphismus liefert die Bedeutungszuordnung von der Syntax in die semantische Algebra.

Beispiel (zur „algebraischen Semantik")

Wir knüpfen mit diesem Beispiel an unser Einführungsbeispiel 1.2 an, in dem wir eine konkrete Syntax einer sehr einfachen Programmiersprache als heterogene Algebra vorgestellt hatten. Jetzt erkennen wir, daß wir dort die später vorgestellte allgemeine Methode der Zuordnung der syntaktischen Algebra zu einer gegebenen kontextfreien Grammatik benutzt haben. Die syntaktische Algebra der dort vorgestellten einfachen Sprache können wir nun schreiben als

$$\underline{\mathsf{SYN}} = [\mathrm{Ide}, \mathrm{Exp}, \mathrm{Com}\ ;\ (f_p)_{p\in P}],$$

wobei P die Produktionenmenge bzw. die Menge der Namen (zugeordneter „Marken") dieser Produktionen bezeichnet.

An diesem einfachen Beispiel soll das Prinzip der Methoden der "algebraic semantics" demonstriert werden - dabei werden der Kürze wegen sämtliche Schwierigkeiten umgangen, die eine reale Programmiersprache mit sich bringt: so die Beschreibung von Input, Output, Seiteneffekten bei der Auswertung der Ausdrücke, Prozeduren, indizierte Variablen, Iterationen usw. Allerdings bleibt bei diesen Merkmalen die Methode ebenfalls die gleiche, nur die Hilfsmittel werden mehr und komplizierter.

Die semantische Algebra $\underline{\mathsf{SEM}}$ hat die gleiche Signatur wie $\underline{\mathsf{SYN}}$,

$$\underline{\mathsf{SEM}} = [\mathrm{Ide}, \mathrm{Ed}, \mathrm{Cd}\ ;\ (g_p)_{p\in P}],$$

die erste Trägermenge ist die gleiche wie in $\underline{\mathsf{SYN}}$, d. h. die Identifier bezeichnen sich selbst!
Die Bezeichnung Ed bzw. Cd steht für "expression denotations" bzw. "command denotations", zu ihrer Definition und dann der semantischen Operationen g_p sind zuvor noch einige Hilfsbereiche und Hilfsoperationen einzuführen:
Es seien

$$\underline{V} = I\!N \cup \underline{Bool}\,, \quad I\!N = \{0,1,2,\ldots\}\,, \quad \underline{Bool} = \{T,F\} \quad \text{„}\mathit{Wertemengen}\text{“}.$$

$+$ bezeichne die Addition in $I\!N$, $\neg$ die Negation in $\underline{Bool}$, beide Operationen werden in trivialer Weise auf $\underline{V}$ ausgedehnt; dazu sei $\underline{\texttt{err}}$ ein neues Element $\notin \underline{V}$:

$$plus: \quad \underline{V}\times\underline{V} \mapsto \underline{V} \qquad \text{vermöge } plus(v_1,v_2) = \begin{cases} v_1+v_2 & \text{falls } v_1,v_2\in I\!N \\ \underline{\texttt{err}} & \text{sonst} \end{cases}.$$

$$non: \quad \underline{V} \mapsto \underline{V} \qquad \text{vermöge } non(v) = \begin{cases} \neg v & \text{falls } v\in \underline{Bool} \\ \underline{\texttt{err}} & \text{sonst} \end{cases}.$$

Bezüglich $\underline{\texttt{err}}$ mag hier die Vereinbarung gelten, daß der Wert einer beliebigen Funktion wieder $\underline{\texttt{err}}$ ist,

falls nur eines ihrer Argumente $\underline{\texttt{err}}$ ist (die sogenannte „strikte“ Erweiterung aller Funktionen).
Wir führen weiter ein

$$\underline{S} = \text{Ide} \mapsto (\underline{V} \cup \{\underline{\texttt{err}}\}).$$

Hierbei bedeutet die Symbolik $F = M \mapsto N$, daß F die Menge aller Funktionen von M in N ist.
Nun ist

$$\begin{aligned} \text{Ed} &= \underline{S} \mapsto (\underline{V} \cup \{\underline{\texttt{err}}\}), \\ \text{Cd} &= \underline{S} \mapsto (\underline{S} \cup \{\underline{\texttt{err}}\}). \end{aligned}$$

Es bleiben die *semantischen Operationen* zu definieren:

Für alle $s \in \underline{S}$, $I \in \text{Ide}$, $e, e', e_1, e_2 \in \text{Ed}$, $c, c_1, c_2 \in \text{Cd}$:

$$\left.\begin{array}{llllll} g_0 & : \mapsto \text{Ed}, & \text{d. h.} & g_0 \in \text{Ed}, & \text{mit} & g_0(s) = 0 \\ g_1 & : \mapsto \text{Ed}, & \text{d. h.} & g_1 \in \text{Ed}, & \text{mit} & g_1(s) = 1 \\ g_{true} & : \mapsto \text{Ed}, & \text{d. h.} & g_{true} \in \text{Ed}, & \text{mit} & g_{true}(s) = T \\ g_{false} & : \mapsto \text{Ed}, & \text{d. h.} & g_{false} \in \text{Ed}, & \text{mit} & g_{false}(s) = F \end{array}\right\} \text{konstante Funktionen!}$$

$$\begin{array}{llllll} g_I & : \mapsto \text{Ed}, & \text{d. h.} & g_I \in \text{Ed}, & \text{mit} & g_I(s) = s(I) \end{array}$$

$$\begin{array}{lllllll}
g_\neg : & \text{Ed} & \mapsto \text{Ed} & \text{mit} & g_\neg(e) = e' & \text{so daß:} & e'(s) = non(e(s)) \\
g_= : & \text{Ed} \times \text{Ed} & \mapsto \text{Ed} & \text{mit} & g_=(e_1, e_2) = e & \text{so daß:} & e(s) = \begin{cases} T & \text{, falls } e_1(s) = e_2(s) \in \underline{V} \\ F & \text{, falls } e_1(s)\,,\, e_2(s) \in \underline{V}, \\ & \quad e_1(s) \neq e_2(s) \\ \underline{\texttt{err}} & \text{, sonst} \end{cases} . \\
g_+ : & \text{Ed} \times \text{Ed} & \mapsto \text{Ed} & \text{mit} & g_+(e_1, e_2) = e & \text{so daß:} & e(s) = plus(e_1(s)\,,\, e_2 s)) \\
g_{:=} : & \text{Ide} \times \text{Ed} & \mapsto \text{Cd} & \text{mit} & g_{:=}(I, e) = c & \text{so daß:} & c(s) = s\,[\,e(s)\,/\,I\,] \\
g_{if} : & \text{Ed} \times \text{Cd} \times \text{Cd} & \mapsto \text{Cd} & \text{mit} & g_{if}(e, c_1, c_2) = c & \text{so daß:} & c(s) = \begin{cases} c_1(s) & \text{, falls } e(s) = T \\ c_2(s) & \text{, falls } e(s)\,,\, F \\ \underline{\texttt{err}} & \text{, sonst} \end{cases} . \\
g_; : & \text{Cd} \times \text{Cd} & \mapsto \text{Cd} & \text{mit} & g_;(c_1, c_2) = c & \text{so daß:} & c(s) = c_2(c_1(s)).
\end{array}$$

Hierbei bedeutet die Symbolik $f\,[\,y\,/\,x\,]$ für eine Funktion f, daß f an genau der Stelle x abgeändert wird, und zwar wird $f(x)$ durch y ersetzt. Alle anderen Funktionswerte bleiben unverändert. Also,

$$f\,[\,y\,/\,x\,] =_{df} f' \quad mit\ f'(z) = \begin{cases} y & , \quad \text{falls } z = x \\ f(z), & \text{falls } z \neq x \end{cases} .$$

Der eindeutig bestimmte Homomorphismus h : $\underline{\mathsf{SYN}} \mapsto \underline{\mathsf{SEM}}$ mit $h = (h_I, h_e, h_c)$, wobei:

$$\begin{array}{lll} h_I & = & \iota_{\text{Ide}} \quad \text{(Identität über Ide)} \\ h_e & : & \text{Exp} \mapsto \text{Ed} \\ h_c & : & \text{Com} \mapsto \text{Cd} \end{array}$$

legt die Semantik der Sprache fest.

5.4 Wertberechnung und Substitution

In diesem Abschnitt soll nun noch der Zusammenhang hergestellt werden zu Bezeichnungen, wie sie besonders in Logik- und ähnlichen Kalkülen üblich sind.
Dazu betrachten wir eine beliebige Σ-Ausdrucksalgebra $\mathcal{E} = [(E_s)_{s\in S}; (f_\omega)_{\omega\in\Omega}]$ über dem Alphabet X, die als syntaktische Algebra dienen soll. Zu einer vorgegebenen semantischen Algebra $\mathcal{A} = [(A_s)_{s\in S}; (g_\omega)_{\omega\in\Omega}]$ und jeder $\mathcal{A}$-Belegung von X, d.h. $b_s : X_s \mapsto A_s$, existiert nach dem Fortsetzungssatz eindeutig ein Homomorphismus b^* von $\mathcal{E}$ in $\mathcal{A}$, der b fortsetzt.
b stellt eine *Belegung* der *Variablen* von X mit *Werten*aus A dar, b^* damit eine *Auswertung* der Ausdrücke in dem semantischen Bereich $\mathcal{A}$ bei der Belegug b.

Übliche Schreibweisen für diesen *semantischer Homomorphismus* :

$$b^* = \mathrm{wert}(-, b) \quad \text{bzw.} \quad b^*(e) = \mathrm{wert}(e, b) \text{ für } e \in E$$

Wählt man als „semantische Algebra“ wieder die syntaktische Algebra $\mathcal{E}$, so haben wir folgenden Spezialfall: Jede $\mathcal{E}$–Belegung s von X ordnet den Variablen sortenrein Ausdrücke zu, stellt also eine *simultane Einsetzung* von Ausdrücken für Variablen dar. Die „Auswertung“ der Ausdrücke bei einer solchen Einsetzung ist durch deren homomorphe Fortsetzung $s^* : \mathcal{E} \mapsto \mathcal{E}$ gegeben. Die Abbildung s^* beschreibt damit die Wirkung der Einsetzung s auf Ausdrücke und wird deshalb als *Substitutionsendomorphismus* und durch

$$\mathrm{sub}(-, s) \quad \text{bzw.} \quad \mathrm{sub}_s$$

bezeichnet.

Beachte:

Substitution ist in diesem Sinne als ein Spezialfall der Semantik anzusehen.

Satz 5.1 (Wertänderung bei Substitutionen)

$\mathcal{E} = [E, F]$ sei Σ-Ausdrucksalgebra über X und $\mathcal{A} = [A, G]$ eine beliebige Σ-Algebra. $b : X \mapsto A$ sei eine $\mathcal{A}$–Belegung von X und $s : X \mapsto E$ bezeichne eine Einsetzung. Ist dann $b' : X \mapsto A$ die wie folgt durch die Einsetzung s festgelegte $\mathcal{A}$-Belegung mit

$$b'(x) = \mathrm{wert}(s(x), b) \quad \textit{für alle } x \in X,$$

so gilt für alle $e \in E$:

$$\mathrm{wert}(\mathrm{sub}(e, s), b) = \mathrm{wert}(e, b').$$

Beweis

Die Behauptung läßt sich wie folgt umformulieren:

$$\underbrace{\mathrm{wert}(\overbrace{\mathrm{sub}(e, s)}^{s^*(e)}, b)}_{b^*(s^*(e))} = \underbrace{\mathrm{wert}(e, b')}_{b'^*(e)},$$

also

$$b^*(s^*(e)) = b'^*(e).$$

$$\left.\begin{array}{rcl} \mathrm{wert}(-, b) = b^* & : & \mathcal{E} \mapsto \mathcal{A} \\ \mathrm{sub}(-, s) = s^* & : & \mathcal{E} \mapsto \mathcal{E} \\ \mathrm{wert}(-, b') = b'^* & : & \mathcal{E} \mapsto \mathcal{A} \end{array}\right\} \text{Homomorphismen}$$

Nach Vorausetzung gilt:

$$b' = b^* \circ s$$

Damit haben wir für deren homomorphe Fortsetzung (sie existiert, weil $\mathcal{E}$ Ausdrucksalgebra ist)

$$b'^* = (b^* \circ s)^* \text{ als Homomorphismus von } \mathcal{E} \text{ in } \mathcal{A}.$$

Nun ist $b^* \circ s^*$ als Nacheinanderausführung zweier Homomorphismen wieder Homomorphismus, und zwar von $\mathcal{E}$ in $\mathcal{A}$. Außerdem gilt:

$$(b^* \circ s^*)\,|_X = b^* \circ s^*\,|_X = b^* \circ s = b'.$$

Wegen der Eindeutigkeit der homomorphen Fortsetzung ist also

$$b'^* = b^* \circ s^*,$$

was behauptet wurde. q.e.d.

Kapitel 6

Kongruenzrelationen und Faktoralgebren

Definition 6.1

Eine Σ–*Kongruenz* R über einer Σ–Algebra $\mathcal{A}$ ist eine Familie $R = (R_s)_{s\in S}$ von Äquivalenzrelationen R_s in A_s mit der Eigenschaft der *Kompatibilität*, d.h. für alle $\omega \in \Omega$ mit $\alpha(\omega) = (s_1, \ldots, s_n, s)$ gilt:

wenn $(a_i, b_i) \in R_{s_i}$ für $1 \leq i \leq n$, so ist auch

$$(f_\omega(a_1, \ldots, a_n), f_\omega(b_1, \ldots, b_n)) \in R_s.$$

Es sei im folgenden die Äquivalenzklasse eines beliebigen $a \in A_s$ bezüglich der Äquivalenzrelation R_s bezeichnet mit

$$[a]_{R_s} \text{ bzw. } [a]_R \text{ bzw. } [a]_s \text{ bzw. } [a].$$

Also

$$[a]_{R_s} = \{\ a' \ \mid\ a' \in A_s \wedge (a, a') \in R_s\ \}.$$

Weiter sei wie üblich $A/R = (A_s/R_s)_{s\in S}$ die Familie der Mengen von Äquivalenzklassen, wobei

$$A_s/R_s = \{\ [a]_{R_s} \ \mid\ a \in A_s\ \}.$$

Die Familie A/R läßt sich nun für den Fall, daß R Kongruenz ist, leicht zu einer Σ–Algebra ausbauen, weil die ursprünglichen Operationen f_ω der Algebra A in naheliegender Weise Operationen $\overline{f_\omega}$ mit den Äquivalenzklassen induzieren. Wir definieren für $\omega \in \Omega$ mit $\alpha(\omega) = (s_1, \ldots, s_n, s)$

$$\overline{f_\omega}([a_1]_{R_{s_1}}, \ldots, [a_n]_{R_{s_n}}) =_{df} [f_\omega(a_1, \ldots, a_n)]_{R_s}.$$

Wegen der Kompatibilität von R ist diese Definition nämlich repräsentantenunabhängig, und die $\overline{f_\omega}$ sind somit tatsächlich Operationen über A/R:
Gilt nämlich $[a_1]_{R_{s_1}} = [a_1']_{R_{s_1}}, \ldots, [a_n]_{R_{s_n}} = [a_n']_{R_{s_n}}$, d. h. $(a_1, a_1') \in R_{s_1}, \ldots, (a_n, a_n') \in R_{s_n}$, so auch $(\ f_\omega(a_1, \ldots, a_n),\ f_\omega(a_1', \ldots, a_n')\ ) \in R_s$, d. h.

$$[f_\omega(a_1, \ldots, a_n)]_{R_s} = [f_\omega(a_1', \ldots, a_n')]_{R_s}.$$

Definition 6.2

Die Σ–Algebra $[A/R\ ,\ (\overline{f_\omega})_{\omega\in\Omega}]$ mit den wie eben definierten Operationen $\overline{f_\omega}$ heißt die *Faktoralgebra* $\mathcal{A}/R$ der Σ–Algebra $\mathcal{A}$ nach der Σ–Kongruenz R.

Satz 6.1 (Homomorphiesatz)

Es sei $\mathcal{A} = [(A_s)_{s\in S}; (f_\omega)_{\omega\in\Omega}]$ eine Σ–Algebra und $\equiv$ eine Σ–Kongruenz. Dann ist die Abbildung $\varphi : A \mapsto A \,/ \equiv$ mit $\varphi(a) = [a]_{\equiv_s}$ für $a \in A_s$ ein Σ–Homomorphismus von $\mathcal{A}$ nach $\mathcal{A}/\equiv$.
Ist ferner $h : \mathcal{A} \mapsto \mathcal{B}$ ein Σ–Homomorphismus und $\equiv_h$ ist definiert durch

$$a \equiv_h a' \quad \text{genau dann, wenn} \quad h(a) = h(a'),$$

dann ist $\equiv_h$ eine Σ–Kongruenz über $\mathcal{A}$; und wenn h surjektiv ist, dann sind $\mathcal{A}/\equiv_h$ und $\mathcal{B}$ isomorph.

$$\begin{tikzcd}
\mathcal{A} \arrow[r, "h"] \arrow[dr, "\varphi"'] & \mathcal{B} \arrow[d, shift left, start anchor=south west] \\
& \mathcal{A} \,/ \equiv_h \arrow[u, shift left, "\cong"']
\end{tikzcd}$$

Bemerkung

Die obige Abbildung φ heißt *natürliche Abbildung*,

$$\varphi = \mathrm{nat}(\equiv).$$

Die Relation $\equiv_h$ (Bildgleichheit bezüglich h) heißt *Kern* von h

$$\equiv_h = \ker(h).$$

Damit ist in obiger Abbildung

$$\varphi \;=\; \mathrm{nat}(\ker(h))$$

zu setzen.

Beweis des Homomorphiesatzes

1. φ ist eine eindeutige Abbildung von A **auf** $A \,/ \equiv$.
   Weiter

$$\begin{aligned}
\varphi_s(f_\omega(a_1,\ldots,a_n)) &= [f_\omega(a_1,\ldots,a_n)]_{R_s} \\
&= \overline{f_\omega}([a_1]_{R_{s_1}},\ldots,[a_n]_{R_{s_n}}) \\
&= \overline{f_\omega}(\varphi_{s_1}(a_1),\ldots,\varphi_{s_n}(a_n)).
\end{aligned}$$

   Also ist φ ein Homomorphismus von $\mathcal{A}$ auf $\mathcal{A} \,/ \equiv$.

2. Seien $a_1 \equiv_{h_{s_1}} a_1', \ldots, a_n \equiv_{h_{s_n}} a_n'$. Dann gilt $h_{s_i}(a_i) = h_{s_i}(a_i')$ für alle $1 \leq i \leq n$.
   Weiter ist

$$\begin{aligned}
h_s(f_\omega^{\mathcal{A}}(a_1,\ldots,a_n) &\overset{\text{Hom.}}{=} f_\omega^{\mathcal{B}}(h_{s_1}(a_1),\ldots,h_{s_n}(a_n)) \\
&= f_\omega^{\mathcal{B}}(h_{s_1}(a_1'),\ldots,h_{s_n}(a_n')) \\
&\overset{\text{Hom.}}{=} h_s(f_\omega^{\mathcal{A}}(a_1',\ldots,a_n')),
\end{aligned}$$

   das heißt

$$f_\omega^{\mathcal{A}}(a_1,\ldots,a_n) \equiv_h f_\omega^{\mathcal{A}}(a_1',\ldots,a_n'),$$

   also ist $\equiv_h$ eine Σ–Kongruenz.

3. Sei h nun surjektiv. Betrachte $\psi : A \,/ \equiv_h \mapsto B$ mit $\psi_s([a]_{h_s}) = h_s(a)$ für alle $a \in A_s$ und alle $s \in S$.

   (a) ψ eindeutig (repräsentantenunabhängig):
   Wenn $[a]_{\equiv_{h_s}} = [a']_{\equiv_{h_s}}$, so $h_s(a) = h_s(a')$ per def. von $\equiv_h$.

   (b) ψ umkehrbar eindeutig:
   Wenn $\psi_s([a]_{\equiv_{h_s}}) = \psi_s([a']_{\equiv_{h_s}})$, d.h. $h_s(a) = h_s(a')$, so $[a]_{\equiv_{h_s}} = [a']_{\equiv_{h_s}}$ nach Definition.

   (c) ψ ist Abbildung **auf** $\mathcal{B}$:
   h ist surjektiv, d.h. für alle $b \in B$ existiert $s \in S$ und $a \in A_s$ mit $h_s(a) = b$. Also $b = \psi_s([a_s]_{\equiv_{h_s}})$.

   (d) Relationstreue:

$$
\begin{aligned}
\psi_s(\overline{f_\omega}([a_1]_{\equiv_{h_{s_1}}}, \ldots, [a_n]_{\equiv_{h_{s_n}}})) &\overset{\text{Def. Faktoralg.}}{=} \psi_s([f_\omega(a_1, \ldots, a_n)]_{\equiv_{h_s}}) \\
&\overset{\text{Def. } \psi}{=} h_s(f_\omega(a_1, \ldots, a_n)) \\
&\overset{\text{Hom.}}{=} f_\omega^{\mathcal{B}}((h_{s_1}(a_1), \ldots, h_{s_n}(a_n)) \\
&\overset{\text{Def. } \psi}{=} f_\omega^{\mathcal{B}}(\psi_{s_1}([a_1]_{\equiv_{h_{s_1}}}), \ldots, \psi_{s_n}([a_n]_{\equiv_{h_{s_n}}})).
\end{aligned}
$$

q.e.d.

Bemerkung

1. Jede Faktoralgebra ist also homomorphes Bild der Ausgangsalgebra, und umgekehrt ist jedes homomorphe Bild einer Algebra isomorph zu einer Faktoralgebra dieser Algebra.
2. Damit kann *jedes* homomorphe Bild einer Algebra völlig „innerhalb" der Algebra gefunden werden.
3. Eine Faktoralgebra ist völlig festgelegt durch die Algebra und eine Kongruenzrelation. Nach dem Homomorphiesatz kann das Studium der Homomorphismen in gewissem Sinne ersetzt werden durch das Studium der Kongruenzrelationen.
4. Eine semantische Algebra $\underline{\mathsf{SEM}}$ ist per def. ein homomorphes Bild der syntaktischen Algebra $\underline{\mathsf{SYN}}$. Damit gilt: $\underline{\mathsf{SEM}} \cong \underline{\mathsf{SYN}}\,/R$ für eine eindeutig bestimmte Kongruenzrelation R.
   Damit kann man die Semantik als Kongruenzrelation über der Syntax erklären. (Man sage, was äquivalente Wirkung hat.)

Satz 6.2

Sei $\mathcal{A}$ eine Σ–Algebra und R eine Σ–Kongruenz über $\mathcal{A}$. Wenn X ein Erzeugendensystem von $\mathcal{A}$ ist, dann ist X/R ein Erzeugendensystem der Faktoralgebra $\mathcal{A}/R$.

Beweis

Vorausgesetzt ist $A = [\,X\,]_{\mathcal{A}}$. Nach dem Homomorphiesatz ist $\varphi = \mathrm{nat}(R) : A \mapsto A/R$ Homomorphismus von $\mathcal{A}$ **auf** $\mathcal{A}/R$. Also gilt

$$A/R = \varphi(A) = \varphi([X]_{\mathcal{A}}) = [\,\varphi(X)\,]_{\mathcal{A}/R}.$$

entsprechend einem früheren Satz. Es ist aber $\varphi(X) = (\,\varphi_s(X_s)\,)_{s \in S}$ mit

$$\varphi_s(X_s) = \{\,\varphi_s(x) \mid x \in X_s\,\} = \{\,[x]_{R_s} \mid x \in X_s\,\} = X_s/R_s,$$

womit $A/R = [\,X/R\,]_{\mathcal{A}/R}$ gilt. q.e.d.

Kapitel 7

Gleichungsdefinierbare Klassen

Die Standardtermalgebra $\underline{T_\Sigma(X)}$ wird manchmal auch als *anarchische* bzw. *absolut freie* Algebra bezeichnet, weil keinerlei Gesetze in ihr gelten. In vielen bekannten Algebren (man denke etwa an Gruppen, Ringe oder Verbände) gelten aber bestimmte Relationen zwischen ihren Elementen, die oft durch Gleichungen ausdrückbar sind. Umgekehrt lassen sich durch Gleichungen Algebrenklassen festlegen.

Definition 7.1

Eine Σ–*Gleichung* der Sorte S über dem Variablensystem $X = (X_s)_{s \in S}$ ist ein Paar von Ausdrücken (e_1, e_2) aus $T_\Sigma(X)_s$.

Ein *Gleichungssystem* E über X ist eine S–sortige Familie von Mengen von Σ–Gleichungen. Für $(e_1, e_2) \in E$ schreibt man oft auch: $e_1 = e_2$.

Definition 7.2

Es sei $\mathcal{A}$ eine Σ–Algebra[1] . Man sagt, die $\mathcal{A}$–Belegung b von X *erfüllt* die Σ–Gleichung (e_1, e_2) über X, kurz

$$b \ \mathrm{erf}_{\mathcal{A}}\, e_1 = e_2$$

genau dann, wenn für die homomorphe Fortsetzung b^* von b auf $\underline{T_\Sigma(X)}$ gilt:

$$b^*(e_1) = b^*(e_2),$$

d.h. $(e_1, e_2) \in \ker(b^*)$.
Die Gleichung (e_1, e_2) heißt eine *Identität* in $\mathcal{A}$ bzw. in $\mathcal{A}$ *gültig*, kurz

$$\mathcal{A} \models e_1 = e_2$$

genau dann, wenn für alle $\mathcal{A}$–Belegungen b von X gilt: $b \ \mathrm{erf}_{\mathcal{A}}\, e_1 = e_2$.
Ein Σ–Gleichungssystem E heißt *gültig* in $\mathcal{A}$, oder $\mathcal{A}$ ist ein *Modell* für E, kurz

$$\mathcal{A} \models E$$

genau dann, wenn alle Gleichungen von E in $\mathcal{A}$ gültig sind. $\mathcal{A}$ heißt dann auch eine (Σ, E)–*Algebra* .

Bemerkung

Die Klasse aller (Σ, E)-Algebren heißt die durch E definierte *Varietät* bzw. eine *gleichungsdefinierbare Klas-*

[1] Wir setzen der Einfachheit halber im folgenden voraus, daß alle betrachteten heterogenen Algebren **nur nichtleere** Trägermengen besitzen. Im anderen Fall kommt es zu Komplikationen bzgl. der Gültigkeit und Ableitbarkeit von Gleichungen, wie Goguen/Meseguer 1981 entdeckt haben. Um die sonst auftretenden Schwierigkeiten zu beheben, muß der Ableitbarkeitsbegriff technisch etwas komplizierter gestaltet werden, was wir hier vermeiden wollen.

se. [2]

Definition 7.3

E sei ein Σ–Gleichungssystem über X. Man sagt, daß eine Gleichung $e \in E, e = (e_1, e_2)$ *durch Einsetzung aus E hervorgeht* genau dann, wenn es existieren $e'_1, e'_2 \in T_\Sigma(X)$ mit $(e'_1, e'_2) \in E$ und es gibt eine Einsetzung $s : X \mapsto \underline{T_\Sigma(X)}$ mit $e_1 = \mathrm{sub}(e'_1, s)$ und $e_2 = \mathrm{sub}(e'_2, s)$.
$\mathcal{S}(E)$ bezeichnet die Familie aller Gleichungen, die durch Einsetzung aus E hervorgehen.
$\mathcal{S}(E)$ heißt auch *stabiler Abschluß* von E.

Satz 7.1 (Einsetzungsregel)

$\mathcal{A}$ sei eine (Σ, E)-Algebra. Dann gilt auch $\mathcal{A} \models \mathcal{S}(E)$.
Mit anderen Worten: In Modellen von E sind auch die durch Einsetzung aus E hervorgegangenen Gleichungen gültig.

Beweis

$(e_1, e_2) \in E$. Wenn $\mathcal{A} \models E$, dann gilt für alle $\mathcal{A}$–Belegungen b' : $\mathrm{wert}(e_1, b') = \mathrm{wert}(e_2, b')$.
Sei nun $(e'_1, e'_2) \in \mathcal{S}(E)$ beliebig, sowie b eine beliebige $\mathcal{A}$–Belegung von X. Dann gibt es eine Einsetzung $s : X \mapsto \underline{T_\Sigma(X)}$, sodaß $e'_1 = \mathrm{sub}((e_1, s)$ und $e'_2 = \mathrm{sub}((e_2, s)$.
Wegen des Satzes über Wertänderungen bei Substitutionen folgt, wenn b' die durch s abgeänderte Belegung ist:

$$\begin{aligned} \mathrm{wert}(e'_1, b) &= \mathrm{wert}(\mathrm{sub}(e_1, s), b) \\ &= \mathrm{wert}(e_1, b') \\ &= \mathrm{wert}(\mathrm{sub}(e_2, s), b) \\ &= \mathrm{wert}(e'_2, b) \end{aligned}$$

Damit haben wir $\mathcal{A} \models (e'_1, e'_2)$, also auch $\mathcal{A} \models \mathcal{S}(E)$. q.e.d.

Definition 7.4

Die Relation

$$\equiv_E \;=\; \bigcap \{ R \mid \mathcal{S}(E) \subseteq R \;\wedge\; R \text{ ist } \Sigma\text{–Kongruenz über } \underline{T_\Sigma(X)} \}$$

heißt die von E erzeugte Kongruenz.

Bemerkung

$\equiv_E$ ist offensichtlich Σ–Kongruenz über $\underline{T_\Sigma(X)}$.
Sie ist die kleinste Kongruenz, die $\mathcal{S}(E)$ umfaßt. Sie heißt auch *syntaktische Äquivalenz* in der Klasse der (Σ, E)-Algebren.

[2] Varietäten lassen sich auch als diejenigen Algebrenklassen charakterisieren, die gegen Unteralgebrenbildung, homomorphe Bilder und Bildung des direkten Produkts abgeschlossen sind. Das sogenannte 1. Birkhoff–Theorem besagt, daß der Abschluß einer Algebrenklasse $\mathcal{K}$ gegenüber den genannten drei Operatoren (über Algebrenklassen) gerade notwendige und hinreichende Bedingung dafür ist, daß $\mathcal{K}$ Modellklasse eines geeigneten Gleichungssystems, also durch Gleichungen axiomatisierbar, ist.

Bemerkung

Die Relation $\equiv_E$ läßt sich auch einfach „von unten" erzeugen:
Dazu ist folgender *Ableitungsbegriff* für $\vdash_E e_1 = e_2$ geeignet:
Man definiert $\vdash_E e_1 = e_2$:

0. Wenn $(e_1, e_2) \in \mathcal{S}(E)$, so $\vdash_E e_1 = e_2$.

1. reflexiver Abschluß:
Wenn e_1 und e_2 denselben Term aus $T_\Sigma(X)$ bezeichnen, so $\vdash_E e_1 = e_2$.

2. symmetrischer Abschluß:
Wenn $\vdash_E e_1 = e_2$, so $\vdash_E e_2 = e_1$.

3. transitiver Abschluß:
Wenn $\vdash_E e_1 = e_3$ und $\vdash_E e_3 = e_2$, so $\vdash_E e_1 = e_2$.

4. kompatibler Abschluß:
Wenn $\omega \in \Omega$ und

$$\left.\begin{array}{l} \vdash_E t_{11} = t_{12} \\ \vdash_E t_{21} = t_{22} \\ \vdash_E \; \vdots \; = \vdots \\ \vdash_E t_{n1} = t_{n2} \end{array}\right\}, \text{ dann auch } \vdash_E \omega t_{11} \ldots t_{n1} = \omega t_{12} \ldots t_{n2}.$$

$\vdash_E e_1 = e_2$ nur dann, wenn dies auf Grund der aufgeführten Punkte 0., 1., 2., 3. oder 4. gilt.

Man erhält dann

$e_1 \equiv_E e_2$ genau dann, wenn $\vdash_E e_1 = e_2$.

Satz 7.2

E sei Σ–Gleichungssystem über X. Für jede (Σ, E)-Algebra $\mathcal{A}$ gilt dann:
Wenn $e_1 \equiv_E e_2$, dann auch $\mathcal{A} \models e_1 = e_2$.

Zum Beweis

Aus Zeitgründen führen wir den Beweis hier nicht. Der einfachste Weg ist der, daß man zur Definition der syntaktischen Äquivalenz den obigen Ableitungsbegriff benutzt. Dann läßt sich der Beweis einfach über die Stufe der Ableitbarkeit der Gleichung $e_1 = e_2$ führen. Er beruht dann insbesondere auf dem oben als Einsetzungsregel bezeichneten Satz.

Definition 7.5

Das S–sortige System von Äquivalenzrelationen $\triangleq_E$ in der Algebra $\underline{T_\Sigma(X)}$ mit $e_1 \triangleq_E e_2$ genau dann, wenn für alle (Σ, E)-Algebren $\mathcal{A}$ gilt: $\mathcal{A} \models e_1 = e_2$, heißt *semantische Äquivalenz* der Klasse aller (Σ, E)-Algebren.

Folgerung 7.3

Aus der syntaktischen Äquivalenz folgt stets die semantische Äquivalenz:

Wenn $e_1 \equiv_E e_2$, so auch $e_1 \triangleq_E e_2$.

In der Klasse aller (Σ, E)-Algebren existieren auch freie Algebren, d. h. solche, für die ein entsprechender Fortsetzungssatz gilt. Man erhält eine, indem man in der Termalgebra $\underline{T_\Sigma(X)}$ alle jene Terme identifiziert, die sich mit Hilfe des definierenden Gleichungssystems E ineinander umformen lassen. Das heißt, man hat die Faktoralgebra von $\underline{T_\Sigma(X)}$ nach der Kongruenz $\equiv_E$ zu bilden.

Definition 7.6

E sei ein Σ–Gleichungssystem über X. Dann wird gesetzt:

$$\underline{T_{\Sigma,E}(X)} =_{\mathrm{df}} \underline{T_\Sigma(X)} \;/ \equiv_E$$

Satz 7.4

$\underline{T_{\Sigma,E}(X)}$ ist eine (Σ, E)-Algebra.

Beweis

Betrachte eine beliebige $\underline{T_{\Sigma,E}(X)}$–Belegung b von X und deren homomorphe Fortsetzung b^*:

$$b : X \mapsto T_{\Sigma,E}(X),$$

$$b^* : \underline{T_\Sigma(X)} \mapsto \underline{T_{\Sigma,E}(X)}.$$

Dann gilt für beliebige $e \in T_\Sigma(X)$:

$$b^*(e) = [\mathrm{sub}(e, s_b)]_{\equiv_E},$$

wobei s_b eine Einsetzung $s_b : X \mapsto T_\Sigma(X)$ bezeichnet mit $s_b(x) = t$ genau dann, wenn $b(x) = [t]_{\equiv_E}$.

Beweis dessen durch algebraische Induktion:

(IA) Sei $x \in X$ und $b(x) = [e]_{\equiv_E}$. Dann haben wir

$$b^*(x) = b(x) = [e]_{\equiv_E} = [s_b(x)]_{\equiv_E} = [\mathrm{sub}(x, s_b)]_{\equiv_E}.$$

(IS) Sei $\omega \in \Omega$, $\alpha(\omega) = (w, s)$, $(e_1, \ldots, e_n) \in T_\Sigma(X)^w$ und für diese e_i sei die Behauptung erfüllt.

$$\begin{aligned} b^*(\omega e_1 \ldots e_n) &\overset{\text{Hom.}}{=} f'_\omega(b^*(e_1), \ldots, b^*(e_n)) && f'_\omega \text{ Operation in Faktoralgebra } \underline{T_{\Sigma,E}(X)}. \\ &\overset{\text{(IV)}}{=} f'_\omega([\mathrm{sub}(e_1, s_b)]_{\equiv_E}, \ldots, [\mathrm{sub}(e_n, s_b)]_{\equiv_E}) \\ &\overset{\text{Faktoralg.}}{=} [\omega\,\mathrm{sub}(e_1, s_b) \ldots \mathrm{sub}(e_n, s_b)]_{\equiv_E} \\ &\overset{\text{Hom.}}{=} [\mathrm{sub}(\omega e_1 \ldots e_n, s_b)]_{\equiv_E}. \end{aligned}$$

Damit ist die Zwischenbehauptung bewiesen. Für jede beliebige Gleichung $(e_1, e_2) \in E$ gilt deshalb

$$b^*(e_1) = [\mathrm{sub}(e_1, s_b)]_{\equiv_E} = [\mathrm{sub}(e_2, s_b)]_{\equiv_E} = b^*(e_2),$$

denn mit $(e_1, e_2) \in E$ ist $(\mathrm{sub}(e_1, s_b), \mathrm{sub}(e_2, s_b)) \in \mathcal{S}(E)$ und damit auch

$$\mathrm{sub}(e_1, s_b) \equiv_E \mathrm{sub}(e_2, s_b).$$

Da b beliebig und $(e_1, e_2) \in E$ beliebig gewählt war, folgt, daß $\underline{T_{\Sigma,E}(X)}$ Modell von E ist: $\underline{T_{\Sigma,E}(X)} \models E$.

q.e.d.

Als Folgerung ergibt sich der

Satz 7.5 (Vollständigkeitssatz der Gleichungstheorie / 2.Birkhoff–Theorem)

Die syntaktische Äquivalenz $\equiv_E$ fällt mit der semantischen Äquivalenz $\triangleq_E$ zusammen.

Beweis

1. Aus $e_1 \equiv_E e_2$ folgt stets $e_1 \triangleq e_2$, dies wurde schon weiter oben gezeigt.

2. Sei umgekehrt $e_1 \triangleq e_2$. Das heißt für alle (Σ, E)-Algebren $\mathcal{A}$ gilt: $\mathcal{A} \models e_1 = e_2$, speziell also auch $\underline{T_{\Sigma,E}(X)} \models e_1 = e_2$. Das heißt, für alle $\underline{T_{\Sigma,E}(X)}$–Belegungen b von X haben wir $b^*(e_1) = b^*(e_2)$. Speziell also auch für die natürliche Abbildung $\varphi = \mathrm{nat}(\equiv)$ mit $\varphi(e) = [e]_{\equiv_E}$, die Homomorphismus von $\underline{T_\Sigma(X)}$ auf $\underline{T_{\Sigma,E}(X)}$ ist (Homomorphiesatz). φ ist deshalb auch die homomorphe Fortsetzung von $\varphi' : X \mapsto \underline{T_{\Sigma,E}(X)}$ mit $\varphi'(x) = [x]_{\equiv_E}$. Also gilt $\varphi(e_1) = \varphi(e_2)$, das heißt

   $$[e_1]_{\equiv_E} = [e_2]_{\equiv_E} \text{ bzw. } e_1 \equiv e_2.$$

q.e.d.

Satz 7.6 (Fortsetzungssatz)

Sei E ein Σ–Gleichungssystem über X und $\mathcal{A}$ eine beliebige (Σ, E)-Algebra. Dann existiert zu jeder $\mathcal{A}$–Belegung h von X genau ein Homomorphismus $h^\#$ von $\underline{T_{\Sigma,E}(X)}$ in $\mathcal{A}$, für den $h^\#([x]_{\equiv_E}) = h(x)$ für alle $x \in X$ gilt.

Beweis

Beweisidee:

$$\begin{tikzcd}
\underline{T_\Sigma(X)} / \equiv_E \arrow[drr, Rightarrow, "h^\#"] & & \\
\underline{T_\Sigma(X)} \arrow[u, "\mathrm{nat}(\equiv_E)"] \arrow[rr, "h^*"] & & \mathcal{A} \\
X \arrow[u, "\subseteq"] \arrow[urr, "h"'] & &
\end{tikzcd}$$

Zunächst existiert nach dem Fortsetzungssatz für Ausdrucksalgebren genau ein Homomorphismus $h^* : \underline{T_\Sigma(X)} \mapsto \mathcal{A}$, der h fortsetzt. Für dieses h^* und beliebige $e_1, e_2 \in T_\Sigma(X)$ gilt: wenn $[e_1]_{\equiv_E} = [e_2]_{\equiv_E}$, d. h. $e_1 \equiv_E e_2$, so folgt nach dem vorigen Satz $\mathcal{A} \models e_1 = e_2$, da $\mathcal{A}$ (Σ, E)-Algebra ist.
Damit ist für jede $\mathcal{A}$–Belegung h und deren homomorphe Fortsetzung h^*

$$h^*(e_1) = h^*(e_2).$$

Betrachte nun zu der gegebenen $\mathcal{A}$–Belegung h für beliebige $e \in \underline{T_\Sigma(X)}$

$$\psi : \underline{T_{\Sigma,E}(X)} \mapsto \mathcal{A}$$

mit

$$\psi([e]_{\equiv_E}) = h^*(e).$$

Nach der vorangegangenen Überlegung ist damit ψ eine *eindeutige Abbildung* von $T_\Sigma(X) \; / \equiv_E$ in A. Außerdem gilt

$$\psi([x]_{\equiv_E}) = h^*(x) = h(x) \text{ für alle } x \in X.$$

Weiter ist ψ *Homomorphismus* von $\underline{T_\Sigma(X)}\ / \equiv_E$ in $\mathcal{A}$:
Denn, betrachte ein beliebiges $\omega \in \Omega$, und bezeichne f_ω die ω zugeordnete Operation in $\underline{T_\Sigma(X)}$, f'_ω die entsprechende durch f_ω induzierte Operation in der Faktoralgebra $\underline{T_\Sigma(X)}\ / \equiv_E$ und $f^{\mathcal{A}}_\omega$ die entspechende in $\mathcal{A}$, dann gilt:

$$\begin{aligned}
\psi_s(\, f'_\omega(\,[e_1]_{s_1},\ldots,[e_n]_{s_n}\,)\,) &= \psi_s(\,[f_\omega(\,e_1,\ldots,e_n\,)]_s\,)\\
&= h^*_s(\,f_\omega(\,e_1,\ldots,e_n\,)\,)\\
&= f^{\mathcal{A}}_\omega(\,h^*_{s_1}(e_1),\ldots,h^*_{s_n}(e_n)\,)\\
&= f^{\mathcal{A}}_\omega(\,\psi_{s_1}(\,[e_1]_{s_1}\,),\ldots,\psi_{s_n}(\,[e_n]_{s_n}\,)\,).
\end{aligned}$$

Außerdem ist ψ *eindeutig festgelegt*:
Denn, bezeichne $h^\#\ :\underline{T_\Sigma(X)}\ / \equiv_E \mapsto \mathcal{A}$ irgendeinen Homomorphismus mit $h^\#(\,[x]_{\equiv_E}\,) = h(\,x\,)$ für alle $x \in X$. Dann haben wir für die Einschränkungen der Homomorphismen ψ und $h^\#$ auf das Faktorsystem $X / \equiv_E$

$$\psi\,|_{X/\equiv_E} = h^\#\,|_{X/\equiv_E}\ .$$

Außerdem ist $X / \equiv_E$ Erzeugendensystem der Faktoralgebra $\underline{T_\Sigma(X)}\ / \equiv_E$, weil X Erzeugendensystem von $\underline{T_\Sigma(X)}$ ist. Nach dem Satz von der Eindeutigkeit der homomorphen Fortsetzung ist damit $\psi = h^\#$. q.e.d.

Für den Spezialfall

$$\underline{T_{\Sigma,E}(X)} =_{df} \underline{T_\Sigma / \equiv_E} = \underline{T_{\Sigma,E}(\emptyset)}$$

erhält man als

Folgerung 7.7

Die Algebra $\underline{T_{\Sigma,E}(X)}$ ist initiale Algebra in der Klasse aller (Σ,E)–Algebren, d.h. zu jeder (Σ,E)–Algebra $\mathcal{A}$ existiert genau ein Homomorphismus $h_{\mathcal{A}}:\ \underline{T_{\Sigma,E}(X)} \mapsto \mathcal{A}$.

Kapitel 8

Abstrakte Datentypen

Vorbemerkungen zur Initialität

Eine Σ–Algebra $\mathcal{A}$ heißt *initial* in einer Klasse[1] von Σ–Algebren $\underline{C}$ genau dann, wenn für jede Algebra $\mathcal{B}$ genau ein Homomorphismus $h : \mathcal{A} \mapsto \mathcal{B}$ existiert.

Bisher bewiesen wurde:

1. $\underline{T_\Sigma}$ initial in $\underline{\underline{\mathrm{Alg}}}_\Sigma$ (in der Klasse aller Σ–Algebren).
2. $\underline{T_{\Sigma,E}(X)}$ initial in $\underline{\underline{\mathrm{Alg}}}_{\Sigma,E}$ (in der Klasse aller (Σ, E)-Algebren).

Satz 8.1

Wenn $\mathcal{A}$ und $\mathcal{B}$ beide in einer Algebrenklasse $\underline{C}$ initial sind, so sind sie isomorph. Wenn $\mathcal{B}$ isomorph ist zu einer in $\underline{C}$ initialen Algebra $\mathcal{A}$, so ist auch $\mathcal{B}$ initial in $\underline{C}$.

Beweis

Der Beweis ist relativ einfach. Man vergleiche den Satz über die Isomorphie von Peano-Algebren. Aus Zeitgründen lassen wir ihn hier aber weg.

Folgerung 8.2

Die initialen Algebren bilden, falls sie in irgendeiner betrachteten Algebrenklasse überhaupt existieren, stets eine Isomorphieklasse in der entsprechenden Algebrenklasse.

8.1 Zum Begriff des abstrakten Datentyps (ADT)

In der Einleitung der Vorlesung war klar geworden, daß ein (konkreter) Datentyp eine im allgemeinen heterogene Algebra ist. „*Abstraktion*“ (bei ADT) soll hier bedeuten:

absehen von unwesentlichen Details, nur das Prinzip soll dargestellt werden, nicht dessen Realisierung.

[1] Der Begriff der Initialität stammt aus der *Kategorientheorie*, so daß statt einer „Klasse“ genauer eine sogenannte *Kategorie* von Algebren zu betrachten wäre. Für unsere Zwecke kommen wir hier aber ohne einen Ausflug in die Kategorientheorie aus.

ADT soll unabhängig von der Implementierung sein, d.h. der Nutzer bekommt nur die Operationen (mit Konstanten) in die Hand, erfährt, *was* sie bewirken, aber nicht *wie* sie dies tun.

Dies bringt folgende Vorteile:

1. Progamme können modularisiert werden.
2. Neue, zum Beispiel effektivere oder fehlerkorrigierte Implementierung desselben ADT führt nicht zu Änderungen anderer Programmteile.
3. Unterstützung des schrittweisen Entwurf eines Programms.

Damit meint „Abstraktion“ hier dasselbe wie z. B. in der klassischen Algebra bei dem Begriff der „abstrakten Gruppe “ (unabhängig davon, ob die Elemente die reellen orthogonalen $(2,2)$–Matrizen oder die Bewegungen der euklidischen Ebene usw. sind).
Somit sollte man unter „abstrakt“ dasselbe verstehen wie unter „gleich bis auf Isomorphie“. Eine abstrakte Gruppe ist nach Definition eine Isomorphieklasse von Gruppen.

Damit ist der Gedanke naheliegend, einen ADT als Isomorphieklasse konkreter Datentypen zu definieren. Aber wie ist nun ein ADT als Isomorphieklasse anzugeben?

1. Gedanke Angabe eines konkreten Repräsentanten des zu definierenden ADT
(jedes isomorphe Bild ist dann eine andere Darstellung desselben ADT). Ein entscheidender Nachteil ist, daß durch die Angabe eines konkreten Repräsentanten auch eine konkrete Implementation angegeben wird, die man eventuell gar nicht meint. Somit ist dies unzweckmäßig.

2. Gedanke Eine allgemeinere Darstellung suchen, Charakterisierung durch Axiome!
Eine einfache Form von Axiomen sind Gleichungen. Der Vorteil ist, daß ein fundamentierter theoretischer Hintergrund existiert, die gleichungsdefinierbaren Klassen von Algebren.

Durch Angabe eines Gleichungssytems E (zuvor muß natürlich eine Signatur Σ festgelegt werden) wird bekanntlich eine ganze Algebrenklasse $\underline{\underline{\underline{\mathrm{Alg}}}}_{\Sigma,E}$ charakterisiert. Man meint aber nur eine *Isomorphieklasse* daraus als den anzugebenden ADT . Vom algebraischen Standpunkt aus bietet sich sofort die Klasse der initialen Algebren an. Dies führt zu der folgenden Definition:

Definition 8.1 (initiale Semantik)

Ein *abstrakter Datentyp* ADT ist die Isomorphieklasse einer initialen Algebra in einer Klasse von Σ–Algebren.

Bemerkung

Es gibt auch noch andere Möglichkeiten:
Am bekanntesten sind die sogenannten finalen Algebren und die Klasse der sogenannten verhaltensäquivalenten Algebren.

8.2 Spezifikation von abstrakten Datentypen

Definition 8.2

Eine *Spezifikation* $\underline{\mathrm{SPEC}}$ eines abstrakten Datentyps ist ein Paar

$\underline{\mathrm{SPEC}} = [\Sigma, E]$

aus einer Signatur Σ und einem Σ–Gleichungssystem E.

Bemerkung

Die Semantik einer solchen Spezifikation ist dann die Isomorphieklasse der initialen (Σ, E)–Algebren. Da $\underline{T_{\Sigma,E}(X)}$ initial in $\underline{\underline{\text{Alg}}}_{\Sigma,E}$ ist, ist die Existenz gesichert.

Beispiel 1

für eine einfache Spezifikation eines abstrakten Datentyps:

Die natürlichen Zahlen mit Addition und Multiplikation werden spezifiziert:

$$
\begin{array}{lll}
\underline{\underline{\text{NAT}}} = & \underline{\text{sorts}} & I\!N \\
 & \underline{\text{oprs}} & 0{,}1{:}\to I\!N \\
 & & +, \times : I\!N I\!N \to I\!N \\
 & \underline{\text{vars}} & n,\ m :\ I\!N \\
 & \underline{\text{eqns}} & n + 0 = n \\
 & & n + (m + 1)) = (n + m) + 1 \\
 & & 0 \times n = 0 \\
 & & (n + 1) \times m = (n \times m) + m \\
\underline{\text{end}}\ \underline{\underline{\text{NAT}}} & &
\end{array}
$$

Im Rahmen dieser Vorlesung konnten jetzt abschließend nur Hinweise zur Begriffsbildung des ADT und deren Spezifikation gegeben werden. Damit schließt sich der zu Beginn der Vorlesung angefangene Gedankengang zu den Datentypen.

Interessierte Hörer seien zur notwendigen Vertiefung auf Spezialvorlesungen zur algebraischen Spezifikation bzw. über abstrakte Datentypen verwiesen.

Index