

## 4.5 Terme

- $\Sigma$ -Grundterm
  - Signatur: Syntax
  - Algebra: Semantik
  - Terme: Wörter der Sprache
    - Beschreibt syntaktische Struktur auf Ebene der Signatur
    - Interpretation durch die jeweilige Algebra
      - z.B.  $\text{succ}(\text{add}(n1, n2))$
    - Grundterme: Terme ohne Variablen
    - Terme: Terme mit Variablen

## 4.5 Terme

- $\Sigma$ -Grundterm (Definition)

Sei  $\Sigma = (S, F)$  eine Signatur.

Die Familie der Mengen  $T(\Sigma, S)$  von  $\Sigma$ -Grundtermen besteht für alle  $s \in S$  aus der kleinsten Teilmenge  $T(\Sigma, S)$  von Wörtern über dem Alphabet  $(S \cup F \cup \{“, ”, “, ”\})$ ,

für die gilt

Konstanten

- $c \in T(\Sigma, S) \forall c: \rightarrow s \in F$

Operations-  
symbole +  
zusammen-  
gesetze  
Terme

- $f(t_1, \dots, t_n) \in T(\Sigma, S)$   
 $\forall f: s_1 \dots s_n \rightarrow s \in F$  und alle  $t_i \in T(\Sigma, S_i), i \in \{1, \dots, n\}$

## 4.5 Terme

- $\Sigma$ -Grundterm (Beispiel)
  - Sei gegeben  $\Sigma$ -Nat = (S, F) aus vorherigem Beispiel
  - $T(\Sigma\text{-Nat}, \text{Nat})$ 
    - Stufe 0: one
    - Stufe 1: succ(one), add(one,one)
    - Stufe 2: succ(succ(one)), succ(add(one,one)), add(succ(one),one), add(one,succ(one)), add(add(one,one),succ(one)),...
    - Stufe 3: succ(succ(succ(one))),...
    - ....
- Vereinfachte Schreibweise per Rekursion
  - $T(\Sigma\text{-Nat}, \text{Nat}) = \{\text{one}\} \cup \{\text{succ}(n) | n \in T(\Sigma\text{-Nat}, \text{Nat})\} \cup \{\text{add}(n_1, n_2) | \{n_1, n_2\} \subseteq T(\Sigma\text{-Nat}, \text{Nat})\}$

## 4.5 Terme

- Auswertung von  $\Sigma$ -Grundtermen (Definition)

Sei  $\Sigma = (S, F)$  eine Signatur und  $A$  eine  $\Sigma$ -Algebra.  
Die **Auswertung** (Evaluation/Interpretation)  $\text{eval}(A)$  der Terme zur Signatur  $\Sigma$  in  $A$  ist eine Familie von Abbildungen

- $\text{eval}(A) = (\text{eval}(A)_s : T(\Sigma, s) \rightarrow A_s)_{s \in S}$

Sie ist definiert für alle  $s \in S$  durch:

- $\forall c : \rightarrow s \in F : \text{eval}(A)_s(c) = c_A$
- $\forall f : s_1, \dots, s_n \rightarrow s \in F$  und  $\forall t_i \in T(\Sigma, s_i) :$   
 $\text{eval}(A)_s(f(t_1, \dots, t_n)) = f_A(\text{eval}(A)_{s_1}(t_1), \dots, \text{eval}(A)_{s_n}(t_n))$

Konstanten

Operations-  
symbole +  
zusammen-  
gesetzte  
Terme

## 4.5 Terme

- Auswertung von  $\Sigma$ -Grundtermen (Beispiel)

## 4.5 Terme

- Signatur mit Variablen (Definition)
  - Sei  $\Sigma = (S, F)$  eine Signatur.  
Eine Familie  $X = (X_s)_{s \in S}$  von Mengen,  
die durch  $S$  indiziert ist, heißt  $S$ -sortiert ( $S$ -indiziert).  
Deren Elemente heißen Variablen.
  - $\forall s \in S: X_s \cap F = \{\}$
  - Dann heißt  $\Sigma = (S, F, X)$  **Signatur mit Variablen**



## 4.5 Terme

- Auswertung von allgemeinen  $\Sigma$ -Termen (Beispiel)
  - Signatur  $\Sigma$ -Nat, Algebra A
  - $X_{\text{Nat}} = \{n_1, n_2\}$
  - Variablenbelegung  $v: X \rightarrow A$ 
    - $v_{\text{Nat}}(n_1) = 1909$
    - $v_{\text{Nat}}(n_2) = 5$
  - $\text{xeval}(v)_{\text{Nat}}(\text{add}(\text{succ}(0), \text{succ}(n_2)))$ 
    - $= \text{add}_A(\text{xeval}(v)_{\text{Nat}}(\text{succ}(0)), \text{xeval}(v)_{\text{Nat}}(\text{succ}(n_2)))$
    - $= \text{add}_A(\text{succ}_A(\text{xeval}(v)_{\text{Nat}}(0)), \text{succ}_A(\text{xeval}(v)_{\text{Nat}}(n_2)))$
    - $= \text{add}_A(\text{succ}_A(v_{\text{Nat}}(0)), \text{succ}_A(v_{\text{Nat}}(n_2)))$
    - $= \text{add}_A(\text{succ}_A(0), \text{succ}_A(5))$
    - $= \text{add}_A(1, 6)$
    - $= 7$

## 4.6 Algebraische Spezifikation

- Spezifikation (lat.): Auflistung
  - Auflistung syntaktisch formulierbarer Eigenschaften einer Algebra
- Signaturen repräsentieren die einfachste Form der Spezifikation
- Idee: Wie kann die Interpretation einer Signatur durch eine Algebra als Funktion dargestellt werden?
  - Bildung von Formeln über den Termen einer Signatur
- Spezifikationen erweitern also Signaturen um Formeln, die mit Hilfe von Termen syntaktisch formuliert werden
  - Algebraische Gleichungslogik

## 4.6 Algebraische Spezifikation

- $\Sigma$ -Gleichung (Definition) / Grundgleichung
  - Sei  $A$  eine Algebra mit der Signatur  $\Sigma = (S, F, X)$ .  
Für  $t, t' \in T(\Sigma, X)_{s \in S}$  heißt  
 $e \equiv t =_s t'$  eine  $\Sigma$ -Gleichung.
  - $e$  heißt **Grundgleichung**, falls weder  $t$  noch  $t'$  Variablen enthalten
- Beispiel
  - Signatur  $\Sigma$ -Nat,  $\Sigma$ -Algebra  $A$ 
    - $\text{eval}(A)_{\text{Nat}}(\text{zero}) = \text{eval}(A)_{\text{Nat}}(\text{add}(\text{zero}, \text{zero}))$  gültig
    - $\text{eval}(A)_{\text{Nat}}(\text{zero}) = \text{eval}(A)_{\text{Nat}}(\text{one})$  ungültig!
    - $\text{xeval}(\text{ass})_{\text{Nat}}(\text{add}(n_1, n_2)) = \text{xeval}(\text{ass})_{\text{Nat}}(\text{add}(n_2, n_1))$  gültig

## 4.6 Algebraische Spezifikation

- Menge der wohldefinierten Formeln (Definition)
  - Zu einer Algebra mit der Signatur  $\Sigma = (S, F, X)$   
und einer  $S$ -sortierten Variablenmenge  $X$   
wird die **Menge der wohldefinierten Formeln** über der Signatur  $\Sigma$   
mit  $\text{WFF}(\Sigma)$  bezeichnet.

$\text{WFF}(\Sigma)$  ist die kleinste Menge mit folgenden Eigenschaften:

  1. Jede  $\Sigma$ -Gleichung ist in  $\text{WFF}(\Sigma)$ .
  2. Mit  $G, H \in \text{WFF}(\Sigma)$  sind auch  $\neg G, (G \wedge H), (G \vee H) \in \text{WFF}(\Sigma)$ .
  3. Falls  $x \in X_s, G \in \text{WFF}(\Sigma)$ , dann ist  $(\forall x: G), (\exists x: G) \in \text{WFF}(\Sigma)$ .

## 4.6 Algebraische Spezifikation

- Menge der wohldefinierten Formeln (Beispiel)
  - Zur Signatur  $\Sigma$ -Nat enthält  $WFF(\Sigma\text{-Nat})$ 
    - jede  $\Sigma$ -Gleichung aus  $T(\Sigma\text{-Nat}, S)$ , z.B.  $\text{add}(\text{zero}, n) = n$
    - Negationen, Disjunktion und Konjunktionen, z.B.  $\neg (\text{add}(\text{zero}, n) = n)$
    - $\forall x : \text{add}(\text{zero}, x) = x$

## 4.6 Algebraische Spezifikation

- Algebraische Spezifikation (Definition)
  - Sei  $\Sigma = (S, F, X)$  eine Signatur mit Variablen und  $E$  eine Menge von  $\Sigma$ -Gleichungen.  
  
Dann heißt  $SP = \langle \Sigma, E \rangle$  **algebraische Spezifikation**, falls  $E \subseteq WFF(\Sigma)$ .
- Beispiel
  - Algebraische Spezifikation  $\text{Spec-Nat}$  erweitert Signatur  $\Sigma\text{-Nat}$  um die  $\Sigma$ -Gleichungen
    - $\text{equal}_{\text{Nat}}(\text{zero}, \text{zero}) = \text{true}$
    - $\text{equal}_{\text{Nat}}(\text{succ}_{\text{Nat}}(n_1), \text{succ}_{\text{Nat}}(n_2)) = \text{equal}_{\text{Nat}}(n_1, n_2)$
    - $\text{succ}_{\text{Nat}}(n_1) = \text{add}_{\text{Nat}}(n_1, 1)$
    - $\text{succ}_{\text{Nat}}(\text{zero}) = \text{one}$

## 4.6 Algebraische Spezifikation

- Modell (Definition)
  - Sei  $SP = \langle \Sigma, E \rangle$  eine algebraische Spezifikation und  $A$  eine  $\Sigma$ -Algebra.

Dann heißt  $A$  **Modell** von  $SP$  (SP-Algebra), wenn für alle  $\Sigma$ -Gleichungen gilt

- $A \models e$  ( $e$  gültig in  $A$ ) ( $e$  ist  $t = t'$ )
  - d.h.  $\forall \text{ass}: X \rightarrow A: \text{xeval}(\text{ass})(t) = \text{xeval}(\text{ass})(t')$
- Modelle sind heterogene Algebren mit der gegebenen Signatur, die die gegebenen Formeln erfüllen

## 4.6 Algebraische Spezifikation

- Modell (Beispiel)

## 4.6 Algebraische Spezifikation

- **Semantik einer Algebra**

- Alles, was in der Semantik vorkommen soll, kann in der Signatur bezeichnet werden (**no-junk-Prinzip**)
  - Das heißt, eine Algebra, die Elemente in ihren Trägermengen hat, die wir mit Grundtermen nicht bezeichnen können, können wir mit der Signatur nicht „gemeint“ haben, denn sonst hätten wir eine „größere“ Signatur gewählt.
  - Alles was von der Signatur (mittels der Grundterme) nicht erfasst wird, wird als „junk“ bezeichnet.
- Alles, was in der Semantik vorkommen soll, kann eindeutig in der Signatur bezeichnet werden (**no-confusion-Prinzip**)
  - Das heißt, wir interpretieren die durch eine Signatur gegebene Spezifikation minimal. Sie beschreibt nichts „Überflüssiges“. Wäre eine Algebra mit der Signatur „gemeint“ gewesen, die ein Element in einer Trägermenge hat, das Bild zweier verschiedener Grundterme ist, hätte die Signatur „kleiner“ ausfallen sollen.

## 4.6 Algebraische Spezifikation

- **Semantik einer Algebra**

- Beide Sichten (No-Junk und No-Confusion) sind einschränkend!
- „No Junk“-Sicht verbietet beispielsweise die Spezifikation aller Algebren mit Trägermengen, die mächtiger als  $\text{Nat}$  sind, denn alle Termengen sind maximal so mächtig wie  $\text{Nat}$ . Es kann somit keine Signatur geben, die eine surjektive Abbildung  $\text{eval}$  von der Menge der Grundterme in z.B. die Menge der reellen Zahlen ermöglicht.
- „No Confusion“-Sicht verbietet beispielsweise alle Operationen, die auf den Daten arbeiten, statt sie lediglich zu konstruieren.  $\text{Nat}$  mit den üblichen Interpretationen ist demnach z.B. keine „gemeinte“ Algebra zur Signatur  $\{\text{nat}\}$ ,  $\text{zero}$ ,  $\text{succ}$ ,  $+$ . Denn sowohl der Term  $\text{zero}$  als auch  $\text{zero}+\text{zero}$  bezeichnen beide  $0$  in der Standardalgebra der natürlichen Zahlen.
- Hinweis: Das Problem der „No Junk“-Sicht ist mit den diskutierten Mitteln nicht lösbar! Das Problem der „No Confusion“-Sicht ist durch die Mittel der Gleichungsspezifikation in den Griff zu bekommen.

## 4.6 Algebraische Spezifikation

- Semantik einer Algebra

$\Sigma$ -NatB (ohne +)	Nat	INT	NATMOD3
Nat	N	Z	{0,1,2}
Zero: $\rightarrow$ Nat	0	0	0
Succ: Nat $\rightarrow$ Nat	$n \rightarrow n+1$	$i \rightarrow i+1$	$n \rightarrow (n+1) \bmod 3$
Signatur	Standardalgebra	Junk	Confusion

- INT enthält „junk“, da nicht alle Elemente der Trägermenge  $INT_{\text{NAT}} = Z$  von Grundtermen bezeichnet werden können! Alle negativen Zahlen bleiben syntaktisch unbenannt.
- NATMOD3 enthält keinen „junk“, ist aber „confus“, denn verschiedene Grundterme der Signatur werden gleich interpretiert.

## 4.6 Algebraische Spezifikation

- Semantik einer Algebra

- Die terminale Semantik ist durch das terminale Modell gegeben. Gilt eine Gleichung in einem Modell, so gilt sie auch im terminalen Modell. Das terminale Modell ist somit das maximale Modell.

- initiales Modell ..... terminales Modell

Menge der erfüllten  $\subseteq$  .....  $\subseteq$  Menge der erfüllten  
Gleichungen Gleichungen

Weiterführende Betrachtungen basieren auf Grundlagen der Kategorientheorie.  
Vgl. dazu Kap 8 und 11 in Ehrig, Mahr, Cornelius, Große-Rohde, Zeitz

## 4.6 Algebraische Spezifikation

- Widersprüchlichkeit und Redundanz
  - Eine Algebra ist widersprüchlich, wenn sich  $true = false$  beweisen lässt.
  - Eine Algebra ist redundant, wenn die Menge der beweisbaren Aussagen gleich bleibt auch dann, wenn ein Axiom gestrichen wird.
    - Beispiel: Wenn die „String“-Algebra um das Axiom  $append(new(),s) = s$  erweitert wird, dann ist sie redundant, denn dieses Axiom ist ohnehin ableitbar und trägt somit nicht zu zusätzlichen beweisbaren Axiomen bei.

### 4.6.1 Von Datenstruktur zum Modell

- Datenstruktur ist aus formeller Sicht eine Algebra
  - Datenstruktur besteht aus verschiedenen Datenbereichen und einer Menge von Operationen auf diesen Bereichen  
z.B. INT, REAL, NAT, STRING, STACK, QUEUE
1. Aufstellung einer Signatur  $\Sigma$ 
    - Sorten (Datenbereiche) und Operationen (Methoden, Funktionen)
  2. Formulieren von  $\Sigma$  (Grund-)Termen
    - Ermöglichen die Bezeichnung der Elemente der Datenbereich
    - „No-junk“-Prinzip: Eine Algebra soll nur Elemente enthalten, die durch  $\Sigma$ -Terme bezeichnet werden können
      - Stellt unsere Signatur genügend syntaktische Ausdrücke bereit, um alle Elemente des Datenbereiches zu bezeichnen?  
→ evtl. Signatur erweitern
      - Lassen sich alle Elemente der Implementierung (Algebra) durch die Signatur bezeichnen? → evtl. Algebra einschränken

## 4.6.1 Von Datenstruktur zum Modell

### 3. Aufstellen von Gleichungen

- „No-confusion“-Prinzip: Elemente der Datenbereiche lassen sich durch verschiedene Grundterme beschreiben
  - Alles, was nicht explizit als gleich definiert wird, wird verschieden interpretiert
  - Es muss definiert werden, welche Grundterme die gleichen Datenelemente bezeichnen
- Gesucht: eine Menge  $E$  von Gleichungen, um die Gleichheit von Elementen zu beschreiben (wegen „no-confusion“-Prinzip),
  - die in der Algebra gültig ist (Korrektheit von  $E$ ) und
  - die Menge aller im Modell gültigen Grundgleichungen generiert (Vollständigkeit von  $E$ )

## 4.6.1 Von Datenstruktur zum Modell

### 4. Ableiten von Algebren aus Spezifikation

- Aus Signatur kann eine „Grundtermalgebra“ abgeleitet werden (initiale Semantik)
- Aus Spezifikation (Signatur + Gleichungen) kann eine „Quotiententermalgebra“ abgeleitet werden (initiale Semantik)
- Quotientenalgebra ist idelaerwise isomorph zu unserem Modell (Ausgangs-Algebra)

### 5. Homomorphismus und Algebra

- Wenn es eine strukturerhaltende Abbildung (Homomorphismus) von einer Quotiententermalgebra auf eine Algebra  $X$  gibt, dann entspricht  $X$  der Spezifikation
- Strukturelle Induktion als Werkzeug für den Beweis der Existenz einer solchen Abbildung

## 4.2 Ein Beispiel

### Struktur der Spezifikation

**algebra X**

**introduces sorts Y,Z;**

**operations**

op1 : X -> X

op2 : X -> Y

**constrains op1, op2 so that for all**

x,y : X, z: Y

...

**end X;**

## 4.2 Ein Beispiel

**algebra String introduces sorts String, Char, Nat, Bool;**

**operations**

new:  $\rightarrow$  String

append: String x String  $\rightarrow$  String

add: String x Char  $\rightarrow$  String

length: String  $\rightarrow$  Nat

isEmpty: String  $\rightarrow$  Bool

equal: String x String  $\rightarrow$  Bool

**constrains new, append, add, length, isEmpty, equal so that**

**for all** s,s1,s2: String, c: Char

isEmpty(new()) = true

isEmpty(add (s,c)) = false

...

equal (new(), new()) = true

equal (new(), add(s,c)) = false

equal (add(s1,c),add(s2,c)) = equal (s1,s2)

**end X;**