Koalgebren und Koinduktion

Eine leichte Einführung

Hendrik Tews

Institut für Theoretische Informatik, TU Dresden
tews@tcs.inf.tu-dresden.de

Inhalt

pot. unendliche Folgen	$\boldsymbol{X \longrightarrow A \times X \uplus \{*\}}$
Transitionssyteme	$\boldsymbol{S \longrightarrow \mathcal{P}(L \times S)}$
Automaten	$\boldsymbol{S \longrightarrow A \Rightarrow (S \times B)}$
Kripke Modelle	$\boldsymbol{V \longrightarrow \mathcal{P}(\Phi) \times \mathcal{P}(V)}$

Definition 1 *Eine T–**Koalgebra** ist ein Morphismus $\alpha : X \longrightarrow T(X)$. Dabei ist T ein Endofunktor auf einer Kategorie $\mathcal{C}$.*

Beispiel

Wähle die Kategorie der Mengen und totalen Funktionen **Set**.
Seien A und B beliebige Mengen. Definiere

$$T^{A,B}_{\mathsf{Aut}}(S) \quad = \quad A \Rightarrow (S \times B) \quad = \quad \{f \mid f : A \longrightarrow S \times B\}$$

Eine T_{Aut}–Koalgebra $\alpha : S \longrightarrow A \Rightarrow (S \times B)$ ist gegeben durch zwei Funktionen

$$\begin{aligned} \mathsf{out}_\alpha &: \quad S \times A \longrightarrow B \\ \mathsf{next}_\alpha &: \quad S \times A \longrightarrow S \end{aligned}$$

$T^{A,B}_{\mathsf{Aut}}$–Koalgebren *sind* Automaten mit Eingabealphabet A und Ausgabealphabet B.

Kategorientheorie	im gewöhnlichen Leben
Kategorie	Universum (aller Mengen)
Objekt	Menge
Morphismus	Funktion
Produkt $\times$	Kartesisches Produkt
Koprodukt $+$	disjunkte Vereinigung
Exponent $\Rightarrow$	Funktionsraum
$\mathbf{0}$	$\emptyset$
$\mathbf{1}$	$\{*\}$
Funktor	Mengenwertige Abbildung incl. Funktionen (Datentyp)

Zwei Gesetze

$$\frac{X \longrightarrow Y \quad X \longrightarrow Z}{X \longrightarrow Y \times Z} \qquad\qquad \frac{X \longrightarrow Y \Rightarrow Z}{X \times Y \longrightarrow Z}$$

Definition 2 (Kategorie) *Eine Kategorie $\mathcal{C}$ besteht aus*

- *einer Menge von Objekten* $|\mathcal{C}|$
- *einer Menge von Morphismen* $\mathcal{C}(X,Y)$ *für je zwei Objekte* $X, Y \in |\mathcal{C}|$
- *mit Identitäten* $\mathrm{id}_X \in \mathcal{C}(X,X)$ *für alle* $X \in |\mathcal{C}|$
- *und Komposition* $\begin{tikzcd} X \arrow[r, "f"'] \arrow[rr, bend left, "g \circ f"] & Y \arrow[r, "g"'] & Z \end{tikzcd}$ *für alle* $f \in \mathcal{C}(X,Y)$ *und* $g \in \mathcal{C}(Y,Z)$

so dass

- $\mathrm{id}_Y \circ f \quad = \quad f \circ \mathrm{id}_X \quad = \quad f$
- $(f \circ g) \circ h \quad = \quad f \circ (g \circ h)$

Definition 3 (Produkt) *Ein Objekt $X \times Y$ heißt* ***Produkt von X und Y*** *wenn*

- *die Projektionen $\pi_1 : X \times Y \longrightarrow X$ und $\pi_2 : X \times Y \longrightarrow Y$ existieren*
- *es für alle Morphismen $f : Z \longrightarrow X$ und $g : Z \longrightarrow Y$*
  genau einen *Morphismus $\langle f, g\rangle : Z \longrightarrow X \times Y$ gibt,*
  sodass $\pi_1 \circ \langle f, g\rangle = f$ und $\pi_2 \circ \langle f, g\rangle = g$

$$\begin{tikzcd}
& Z \arrow[dl, "f"'] \arrow[d, dashed, "\langle f,g\rangle"] \arrow[dr, "g"] & \\
X & X \times Y \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & Z
\end{tikzcd}$$

Definition 4 (Funktor) *Seien $\mathcal{C}$ und $\mathcal{D}$ zwei Kategorien.*
Ein Funktor $F : \mathcal{C} \longrightarrow \mathcal{D}$ besteht aus

- *einer Abbildung der Objekte* $F : |\mathcal{C}| \longrightarrow |\mathcal{D}|$
- *einer Abbildung der Morphismen* $F : \mathcal{C}(X, Y) \longrightarrow \mathcal{D}(F(X), F(Y))$ *für alle* $X, Y \in |\mathcal{C}|$

sodass

- $F(\mathrm{id}_X) = \mathrm{id}_{F(X)}$
- $F(f \circ g) = F(f) \circ F(g)$

Beispiel

$$\begin{aligned}
\mathsf{List} &: \mathbf{Set} \longrightarrow \mathbf{Set} \qquad \text{Endofunktor auf } \mathbf{Set} \\
\mathsf{List}(X) &= \{l \mid l \text{ ist eine endliche Liste mit Elementen aus } X\} \\
f &: X \longrightarrow Y \\
\mathsf{List}(f) &: \mathsf{List}(X) \longrightarrow \mathsf{List}(Y) \\
[x_1; x_2; \ldots; x_n] &\longmapsto [f\, x_1; f\, x_2; \ldots; f\, x_n]
\end{aligned}$$

… sind für die Beispiele aus der Informatik meist ausreichend:

$$F(X) \quad = \quad \begin{cases} A \\ X \\ F_1(X) \times F_2(X) \\ F_1(X) + F_2(X) \\ A \Rightarrow F_1(X) \\ \mathcal{P}(F_1(X)) & \text{kovariant – manchmal} \end{cases}$$

Beispiel

$$\begin{array}{rcl} T^{A,B}_{\mathsf{Aut}}(X) & = & A \Rightarrow (X \times B) \\ f & : & X \longrightarrow Y \\ T^{A,B}_{\mathsf{Aut}}(f) & : & T^{A,B}_{\mathsf{Aut}}(X) \longrightarrow T^{A,B}_{\mathsf{Aut}}(Y) \\ h : A \longrightarrow X \times B & \mapsto & \lambda a : A \,.\, (f(\pi_1(h\,a)), \pi_2(h\,a)) \end{array}$$

	List[A]	Aut[A,B]
Charakterisierung	Konstruktoren	Destruktoren
Signatur	$\mathsf{nil} : \mathbf{1} \longrightarrow \mathsf{List}(A)$ $\mathsf{cons} : A \times \mathsf{List}(A) \longrightarrow \mathsf{List}(A)$	$\mathsf{out} : \mathsf{Aut}(A,B) \times A \longrightarrow B$ $\mathsf{next} : \mathsf{Aut}(A,B) \times A \longrightarrow \mathsf{Aut}(A,B)$
Funktor $T(X) =$	$\mathbf{1} + A \times X$	$A \Rightarrow (B \times X)$
Struktur	Algebra $\mathbf{1} + A \times X \longrightarrow X$	Koalgebra $X \longrightarrow A \Rightarrow (B \times X)$
Semantik	initial (freie erzeugte Algebra)	final (Verhalten)
Beweisprinzip	Induktion	Koinduktion
	Gleichheit	Bisimulation

Definition 5 *Seien $c : X \longrightarrow T(X)$ und $d : Y \longrightarrow T(Y)$ zwei T–Koalgebren. Ein Morphismus $f : X \longrightarrow Y$ ist ein Koalgebra-Morphismus $c \longrightarrow d$ falls*

$$T(f) \circ c \quad = \quad d \circ f$$

bzw. diagrammatisch

$$\begin{tikzcd} X \arrow[r, "c"] \arrow[d, "f"'] & T(X) \arrow[d, "T(f)"] \\ Y \arrow[r, "d"] & T(Y) \end{tikzcd}$$

Alle Koalgebren für einen Funktor formen natürlich eine Kategorie!

Beispiel

Seien $c : X \longrightarrow T^{A,B}_{\mathsf{Aut}}(X)$ und $d : Y \longrightarrow T^{A,B}_{\mathsf{Aut}}(Y)$ zwei Automaten-Koalgebren.
Eine Funktion $f : X \longrightarrow Y$ ist ein Automaten-Koalgeben-Morphismus, wenn

$$\begin{aligned} \mathsf{out}_d(f\,x, a) &= \mathsf{out}_c(x, a) \\ \mathsf{next}_d(f, x, a) &= f(\mathsf{next}_c(x, a)) \end{aligned}$$

Seien $c : X \longrightarrow T(X)$ und $d : Y \longrightarrow T(Y)$ zwei T–Koalgebren.

Definition 6 (Aczel / Mendler [1]) *Eine Relation $R \subseteq X \times Y$ ist eine Bisimulation zwischen c und d falls es eine Koalgebra $r : R \longrightarrow T(R)$ gibt, sodass die Projektionen $\pi_1 : R \longrightarrow X$ und $\pi_2 : R \longrightarrow Y$ Koalgebra-Morphismen sind:*

$$
\begin{tikzcd}
X \arrow[d, "c"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "r"] & Y \arrow[d, "d"] \\
T(X) & T(R) \arrow[l, "T(\pi_1)"'] \arrow[r, "T(\pi_2)"] & T(Y)
\end{tikzcd}
$$

Definition 7 (Hermida / Jacobs [2]) *Eine Relation $R \subseteq X \times Y$ ist eine Bisimulation zwischen c und d falls für alle $x \in X$ und $y \in Y$:*

$$x\,R\,y \quad \Longrightarrow \quad (c\,x) \;\; \mathrm{Rel}(T)(R) \;\; (d\,y)$$

Dabei ist $\mathrm{Rel}(T)(R)$ das Relationenlifting von R in bezug auf T.

[1] P. Aczel and P. F. Mendler. A final coalgebra theorem. In D. H. Pitt, D. E. Rydeheard, P. Dybjer, A. M. Pitts, and A. Poigné, editors, *Proceedings of the Conference on Category Theory and Computer Science*, volume 389 of *LNCS*, pages 357–365. Springer, 1989.

[2] C. Hermida and B. Jacobs. Structural induction and coinduction in a fibrational setting. *Information and Computation*, 145(2):107–152, 1998.

Beispiel

Seien $c : X \longrightarrow T^{A,B}(X)$ und $d : Y \longrightarrow T^{A,B}(Y)$ zwei Automaten-Koalgebren.
Eine Relation $R \subseteq X \times Y$ ist eine Automaten-Bisimulation falls für alle $(x, y) \in R$:

- $\mathsf{out}_c(x, a) \quad = \quad \mathsf{out}_d(y, a)$
- $\mathsf{next}_c(x, a) \quad R \quad \mathsf{next}_d(y, a)$

Angenommen T erhält schwache Pullbacks (gilt in der Regel):

Proposition 8

- *Bisimulationen sind unter Vereinigung und Komposition abgeschlossen.*
- *Die Gleichheitsrelation ist immer eine Bisimulation.*
- *Es gibt eine größte Bisimulation; sie heißt **Bisimilarität.***

Relationenlifting (für einen polynomialen Funktor T)

ist eine Operation zwischen Relationen

$$\big(R \quad \subseteq \quad X \times Y\big) \quad \longmapsto \quad \big(\mathsf{Rel}(T)(R) \quad \subseteq \quad T(X) \times T(Y)\big)$$

definiert durch Induktion über die Struktur von T:

$T(X)$	$\mathsf{Rel}(T)(R)(u,v)$ für $u \in T(X)$ und $v \in T(Y)$
X	$R(u,v)$
A	$u =_A v$
$T_1(X) \times T_2(X)$	$\mathsf{Rel}(T_1)(R)(\pi_1\, u, \pi_1\, v) \;\wedge\; \mathsf{Rel}(T_2)(R)(\pi_2\, u, \pi_2\, v)$
$T_1(X) + T_2(X)$	$\begin{cases} \text{falls } (u,v) = (\kappa_1\, u_1, \kappa_1\, v_1) & \mathsf{Rel}(T_1)(R)(u_1, v_1) \\ \text{falls } (u,v) = (\kappa_2\, u_2, \kappa_2\, v_2) & \mathsf{Rel}(T_2)(R)(u_2, v_2) \\ \text{ansonsten} & \perp \end{cases}$
$A \Rightarrow T_1(X)$	$\forall a : A \,.\, \mathsf{Rel}(T_1)(R)(u(a), v(a))$

Definition 9 *Eine Koalgebra* $z : Z \longrightarrow T(Z)$ *ist* ***final****, falls es für jede Koalgebra* $c : X \longrightarrow T(X)$ *genau einen Koalgebra Homomorphismus* $!_c : c \longrightarrow z$ *gibt.*

Finale Koalgebra für Automaten

Trägermenge : $Z \quad = \quad A^+ \Rightarrow B \quad = \quad \{f \mid f : A^+ \longrightarrow B\}$

Koalgebra:

$$\begin{aligned} \mathsf{out}_Z(f, a) &= f(\langle a \rangle) \\ \mathsf{next}_Z(f, a) &= \lambda \langle a_1, \ldots a_n \rangle : A^+ \,.\, f(\langle a, a_1, \ldots a_n \rangle) \end{aligned}$$

Homomorphismus : Für einen Automaten c ergibt sich

$$\begin{aligned} !_c &: S \longrightarrow A^+ \Rightarrow B \\ !_c(s) &= \lambda \langle a_1, \ldots, a_n \rangle : A^+ \,.\, \mathsf{out}_c\big(\mathsf{next}_c\big(\cdots(\mathsf{next}_c(s, a_0)), \ldots, a_{n-1}\big), a_n\big) \end{aligned}$$

Verhaltenstypen Der Verhaltenstyp $\mathsf{Aut}[A,B]$ der Automaten wird definiert durch die finale $T^{A,B}_{\mathsf{Aut}}$–Koalgebra.

Koinduktion = Ausnutzen der Finalität

Automatenverkettung Gegeben sind zwei Automaten c und d:

$$A \longrightarrow \boxed{\quad s \quad} \longrightarrow B \qquad\qquad B \longrightarrow \boxed{\quad q \quad} \longrightarrow C$$

Eigenschaften der Verkettung $d \circ c$:

$$\begin{aligned} \mathsf{out}_{d\circ c}((q \circ s),\ a) &= \mathsf{out}_d(q, \mathsf{out}_c(s,a)) \\ \mathsf{next}_{d\circ c}((q \circ s),\ a) &= \mathsf{next}_d(q, \mathsf{out}_c(s,a)) \circ \mathsf{next}_c(s,a) \end{aligned}$$

definieren eine $T^{A,C}_{\mathsf{Aut}}$–Koalgebra auf $\mathsf{Aut}[B,C] \times \mathsf{Aut}[A,B]$.

Finalität liefert $\quad - \circ - \ : \ \mathsf{Aut}[B,C] \times \mathsf{Aut}[A,B] \longrightarrow \mathsf{Aut}[A,C]$.

Theorem 10 *Die Bisimilarität für die finale Koalgebra ist in der Gleichheit enthalten.*

Um $x = y$ zu beweisen, ist es ausreichend, eine Bisimulation R mit $R(x, y)$ zu finden.

Für jedes Alphabet A gibt es einen Automaten id_A mit genau einem Zustand und $\mathrm{out}_{\mathrm{id}_A}(s, a) = a$.

Lemma 11 *Für die Automatenverkettung gilt:*

$$\mathit{id}_A \circ c \quad = \quad c$$

Beweis: Eine geeignete Bisimulation ist R mit

$$R(x, y) \quad \text{genau dann, wenn} \quad x = \mathrm{id}_A \circ y$$

□

[1] S. Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 2nd edition, 1997. (1st ed., 1971).

[2] M. Barr and C. Wells. *Category Theory for Computing Science*. Prentice Hall Intl. Series in Computer Science. Prentice Hall, London, 2nd edition, 1995. (1st ed., New York, 1990).

[3] R. F. C. Walters. *Categories and Computer Science*. Cambridge University Press, August 1992.

[4] B. Jacobs and J. Rutten. A tutorial on (co)algebras and (co)induction. *EATCS Bulletin*, 62:222–259, 1997.

[5] B. Jacobs. Exercises in coalgebraic specification. In R. Crole R. Backhouse and J. Gibbons, editors, *Algebraic and Coalgebraic Methods in the Mathematics of Program Construction*, volume 2297 of *Lecture Notes in Computer Science*, pages 237–280. Springer, Berlin, 2002.

[6] H. P. Gumm. Elements of the general theory of coalgebras. LUATCS '99, available at www.mathematik.uni-marburg.de/∼gumm/Papers/publ.html, 1999.

[7] J. J. M. M. Rutten. Universal coalgebra: A theory of systems. *Theoretical Computer Science*, 249(1):3–80, October 2000.

[8] H. Reichel. An approach to object semantics based on terminal co–algebras. *Mathematical Structure in Computer Science*, 5:129–152, 1995.

CCSL = Algebren + Coalgebren + HOL

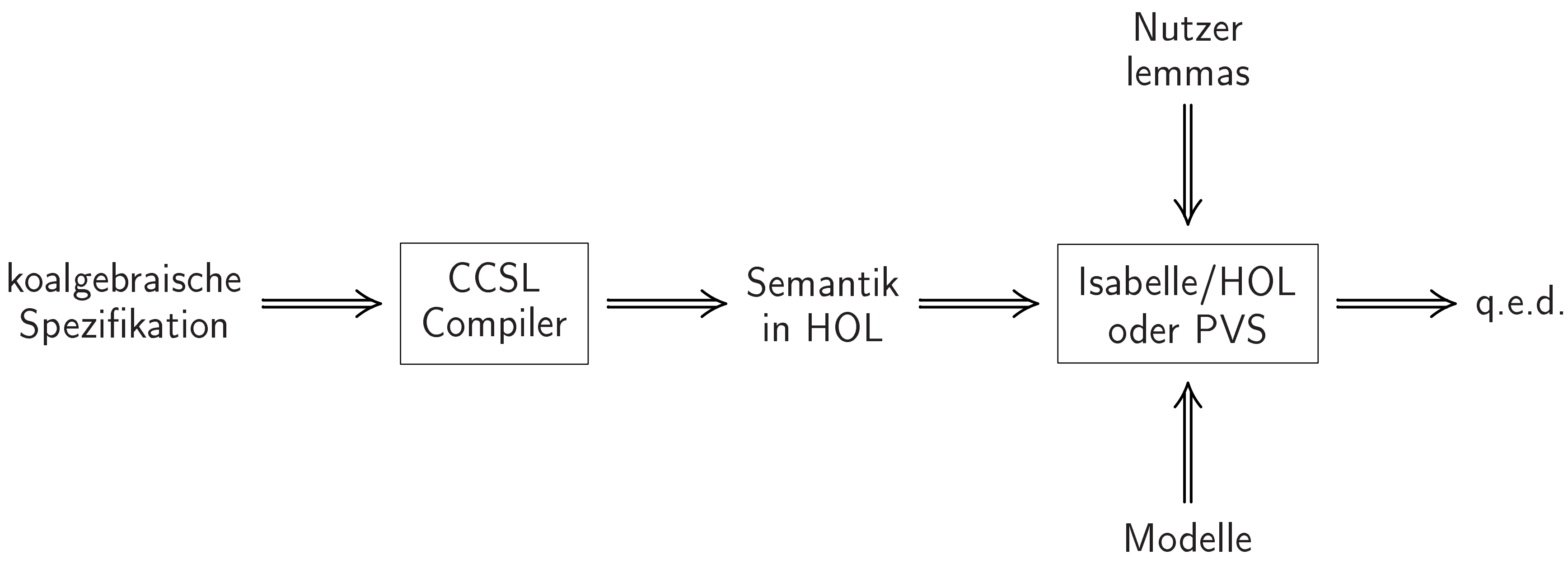
Eine Klassenspezifikation $\mathcal{A}$ besteht aus

- Signatur
  - Methoden Deklarationen
  - Konstruktor Deklarationen
- Axiomen
  - Methodenaxiomen $\mathcal{A}_M$
  - Konstruktoraxiomen $\mathcal{A}_C$
- syntaktischem Zucker
  - Vererbungsklauseln (mit Umbenennung)
  - definitorische Erweiterungen
  - Theoreme
  - Importklauseln
  - Lifting Requests

Die Logik für die Axiome ist signaturabhängig.

- Logik höherer Ordnung (HOL)
- Verhaltensgleichheit
- (methodenweise) modale Operatoren $\Box$ (always) und $\Diamond$ (eventually)

CCSL Compiler



Implementation

- in Ocaml implementiert
- einfachste Compilertechnik, > 10 Pässe
- 40.000 Zeilen, 100 Dateien
- Version 2.2
- frei verfügbar unter http://wwwtcs.inf.tu-dresden.de/~tews/ccsl