Vorlesung vom 23.01.06: Z für Fortgeschrittene

Wo sind wir?

- Teil I: Grundlagen formaler Logik
- Teil II: Modellierung
- Teil III: Spezifikationsformalismen
  - Axiomatische Spezifikation
  - Spezifikation mit Z: Grundlagen & Einführung
  - *Z: Schemakalkül & Verfeinerung*
  - Datenverfeinerung und Verhaltensgleichheit

Schema

- *Schema* sind *Kurznotation*
- Schreibweise:

$$\begin{array}{|l} \underline{\ Name\ } \\ Declaration \\ \hline Predicate \\ \hline \end{array} \qquad Name \mathrel{\widehat=} [\, Declaration \mid Predicate \,]$$

  - Bindet $Name$ an das Schema
  - Schema haben eigenen *Namensraum*
  - *Abkürzende Schreibweise* (keine *Semantik*)

Schema als Typen

- Bekannte *Typen*: *Deklaration*, freier Typ, Potenzmenge, Tupel
- *Schematyp*: benannte Tupel (*binding*)

$$\begin{array}{|l} \underline{\ SchemaOne\ } \\ a : \mathbb{Z} \\ b : \mathbb{P}\,\mathbb{Z} \\ \hline \end{array} \qquad \langle\!|\ a : \mathbb{Z}, b : \mathbb{P}\,\mathbb{Z}\ |\!\rangle$$

  - Vgl. *Tupel* $\langle\!|\ a : \mathbb{Z}, b : \mathbb{P}\,\mathbb{Z}\ |\!\rangle \cong \mathbb{Z} \times \mathbb{P}\,\mathbb{Z}$
  - Reihenfolge irrelevant: $\langle\!|\ a : \mathbb{Z}, b : \mathbb{P}\,\mathbb{Z}\ |\!\rangle = \langle\!|\ b : \mathbb{P}\,\mathbb{Z}, a : \mathbb{Z}\ |\!\rangle$
  - *Elemente*: $\langle\!|\ a \rightsquigarrow 2, b :\rightsquigarrow \{3, 4\}\ |\!\rangle \in SchemaOne$
  - *Selektion* mit ., z.B. $S.a = 2$

Schema als Deklarationen

- Jedes Schemata ist eine *Deklaration*

$$\{SchemaOne \mid a \in b\} \qquad \{a : \mathbb{Z}, b : \mathbb{P}\,\mathbb{Z} \mid a \in b\}$$

- *Charakterisierende Bindung* (*characteristic binding*) θS

$$\theta SchemaOne = \langle\!|\ a \rightsquigarrow a, b \rightsquigarrow b\ |\!\rangle$$

  - Nur *definiert* in *Kontext*, in dem a und b *deklariert* sind

Schema als Prädikate

- Jedes Schema ist ein *Prädikat*:
  - *Allquantifiziert* über *Deklarationen*,
  - *Konjunktion* der *Prädikate*

$$\begin{array}{|l} \underline{\ SchemaTwo\ } \\ SchemaOne \\ \hline a \in b \\ a \geq 10 \\ \hline \end{array} \qquad \forall\, a : \mathbb{Z}, b : \mathbb{P}\,\mathbb{Z} \bullet a \in b \land a \geq 10$$

- *Normalisierung* von Schema: alle *Einschränkungen* in *Prädikaten*

Umbenennung

- Komponenten können *umbenannt* werden: $Schema[new/old]$

$$OtherSchema \mathrel{\widehat=} SchemaTwo[x/a, y/b]$$

$$\begin{array}{|l} \underline{\ OtherSchema\ } \\ x : \mathbb{Z} \\ y : \mathbb{P}\,\mathbb{Z} \\ \hline x \in y \land x \geq 10 \\ \hline \end{array}$$

Generische Schema

- *Parametrisierte* Schemata:

$$\begin{array}{|l} \underline{\ SchemaThree\,[X]\ } \\ a : X \\ b : \mathbb{P}\,X \\ \hline a \in b \\ \hline \end{array}$$

- X steht für *beliebigen Typ*
- Instantiierung: $SchemaThree[\mathbb{Z}]$

- Analog: *generische Definitionen*

$$
\begin{array}{|l}
\hline [X, Y] \\
\mathit{first} : X \times Y \to X \\
\hline \forall X : X, y : Y \bullet \mathit{first}(x, y) = x \\
\hline
\end{array}
$$

$$
\begin{array}{|l}
\hline [X] \\
\_ \frown \_ : \mathsf{seq}\, X \times \mathsf{seq}\, X \to \mathsf{seq}\, X \\
rev : \mathsf{seq}\, X \to \mathsf{seq}\, X \\
\hline \forall s, t : \mathsf{seq}\, X \bullet s \frown t = s \cup \{n : \mathsf{dom}\, t \bullet n + \#s \mapsto t(n)\} \\
\forall s : \mathsf{seq}\, X \bullet revs = (\lambda\, n : \mathsf{dom}\, s \bullet s(\#s - n + 1)) \\
\hline
\end{array}
$$

Der Schemakalkül

Die Operationen im Überblick:

- Konjunktion, Disjunktion und Negation
- Quantifikation und Restriktion
- Dekoration, Δ und Ξ
- Komposition
- Pipes

Konjunktion und Disjunktion

- $S \wedge T$: *Vereinigung* der Deklaration, *Konjunktion* der Prädikate
  - In beiden Schema deklarierte *Variablen*: *gleicher* Typ
- $S \vee T$: *Vereinigung* der Deklaration, *Disjunktion* der Prädikate
  - In beiden Schema deklarierte *Variablen*: *gleicher* Typ
- $\neg\, S$: Deklaration *unverändert*, *Negation* der Prädikate *in toto*
  - Meist *nicht* so sinnvoll

Quantifizierung und Restriktion

- $\forall\, x_1 : T_1 \ldots x_n : T_n \bullet [\, Decl \mid P\,]$:
  - Variablen $x_1, \ldots x_n$ aus $Decl$ entfernen
  - P ersetzen durch $\forall\, x_1 : T_1, \ldots, x_n : T_n \bullet P$
  - Deklaration in $Decl$: *gleicher Typ*
- $\exists\, x_1 : T_1 \ldots x_n : T_n \bullet [\, Decl \mid P\,]$: analog
  - Erlaubt *verdecken* von Bezeichnern
- Abkürzende Notation: $S \setminus L$

$$SchemaTwo \setminus (a) = [\, b : \mathbb{P}\,\mathbb{Z} \mid \exists\, a : \mathbb{Z} \bullet a \in b \wedge a \geq 10\,]$$

Dekoration, Δ und Ξ

- S': wie S, aber alle Deklarationen und Prädikate mit $'$
- $S?, S!$: analog
- Weitere Abkürzungen:

$$\Delta Schema \mathrel{\widehat=} [\, Schema;\ Schema'\,]$$

$$\Xi Schema \mathrel{\widehat=} [\, \Delta Schema \mid \theta Schema = \theta Schema'\,]$$

- Ξ kann *überschrieben* werden

Komposition

- Schema = Zustandsübergang = Operation, dann Komposition von Operationen
- *Voraussetzung*: Für jede *gestrichene* Deklaration in *Quelle ungestrichene* Deklaration in *Ziel*

$$
\begin{aligned}
S \mathbin{⨾} T \mathrel{\widehat=} \exists\, State'' \bullet\ & (\exists\, State' \bullet [\, S;\ State' \mid \theta State' = \theta State''\,]) \wedge \\
& (\exists\, State \bullet [\, T;\ State \mid \theta State = \theta State''\,])
\end{aligned}
$$

Pipes

- Pipe: Verknüpfung der *Ausgabe* mit der *Eingabe*
- *Voraussetzung*:
  Für jede *Ausgabevariable* in *Quelle Eingabevariable* in *Ziel*

$$
\begin{aligned}
S \gg T \mathrel{\widehat=} \exists\, Pipe!? \bullet\ & (\exists\, Pipe! \bullet [\, S;\ Pipe!? \mid \theta Pipe! = \theta Pipe!?) \wedge \\
& (\exists\, Pipe? \bullet [\, T;\ Pipe!? \mid \theta Pipe? = \theta Pipe!?)
\end{aligned}
$$

Zusammenfassung

- *Schema*: abkürzende Schreibweise für . . .
  - *Typen*
  - *Deklarationen*
  - *Prädikate*
- *Schema* können *Systemzustände* modellieren
- *Schemakalkül*:
  - Log. *Operatoren*
  - *Zustandsübergänge*, Δ, Ξ
  - *Komposition*
- Nächste Woche: *Programmentwicklung*