Axiomatische Semantik für **IMP**

- Abstraktionsgrad zwischen operationaler und denotationeller Semantik
- Zweck:
  - Verifikation von Programmen
  - Beweis über *Programmabläufe*
- Idee: Annotierte Programme

Axiomatische Semantik: Beispiel

- Zum Beispiel Fakultät:

```
M:= 1;
```

$$\{M = 1, N = n\}$$

```
while 0 < N do M:= N* M; N:= N- 1 done
```

$$\{N \leq 0 \land M = n!\}$$

Axiomatische Semantik: Notation

- *Vor-* und *Nachbedingungen:*

  $$\{A\}c\{B\}$$

  Bedeutung: jeder Zustand, der A erfüllt, ist nach Termination von c ein Zustand, der B erfüllt.

- *Partielle* und *totale* Korrektheit.

- Zu definieren: Sprache für Vor- und Nachbedingungen (Zusicherungen), Erfüllbarkeit für Zusicherungen.

Zusicherungen

- Zusicherungen: **Assn**

$$\begin{array}{rl} A ::= & \top \mid \bot \mid a_0 = a_1 \mid a_0 \leq a_1 \mid \\ & A_0 \wedge A_1 \mid A_0 \vee A_1 \mid \neg A \mid A_0 \Rightarrow A_1 \mid \forall\, i.A \mid \exists\, i.A \end{array}$$

  Arithmetische Ausdrücke **Aexp** erweitert um Integer-Variablen (**Intvar**).

- Beispiele:
  - m teilt n: $\exists\, p.\, n = p * m$
  - n Primzahl: $\neg(n = 1) \wedge \forall\, m.\, ((\exists\, p.\, n = p * m) \Rightarrow (m = n \vee m = 1))$

- Substitution: $A[a/i]$ (Vorsicht bei gebundenen Variablen!)

Semantik von Zusicherungen

$\sigma \models^I A$: Zustand $\sigma : \Sigma \to \Sigma_\perp$ erfüllt Zusicherung A unter *Variablenbelegung* I : **Intvar** $\to \mathbb{N}$.

$$
\begin{array}{l}
\sigma \models^I \top \\
\sigma \models^I a_0 = a_1 \text{ if } \mathcal{E}[\![a_0]\!]I\sigma = \mathcal{E}[\![a_1]\!]I\sigma \\
\sigma \models^I a_0 \leq a_1 \text{ if } \mathcal{E}[\![a_0]\!]I\sigma \leq \mathcal{E}[\![a_1]\!]I\sigma \\
\sigma \models^I A \wedge B \text{ if } \sigma \models^I A \text{ and } \sigma \models^I B \\
\sigma \models^I A \vee B \text{ if } \sigma \models^I A \text{ or } \sigma \models^I B \\
\sigma \models^I \neg A \text{ if not } \sigma \models^I A \\
\sigma \models^I A \Rightarrow B \text{ if not } \sigma \models^I A\text{, or } \sigma \models^I B \\
\sigma \models^I \forall\, i.A \text{ if } \sigma \models^{I[n/i]} A \text{ for all } n \in \mathbb{N} \\
\sigma \models^I \exists\, i.A \text{ if } \sigma \models^{I[k/i]} A \text{ for some } k \in \mathbb{N} \\
\perp \models^I A
\end{array}
$$

Partielle Korrektheit und Gültigkeit formal

- Partielle Korrektheit:

$$\sigma \models^{I} \{A\}c\{B\} \Longleftrightarrow (\sigma \models^{I} A \Rightarrow \mathcal{D}[\![c]\!]\sigma \models^{I} B)$$

- Gültigkeit:

$$\models \{A\}c\{B\} \Longleftrightarrow \text{ für alle } \sigma, I\text{: } \sigma \models^{I} \{A\}c\{B\}$$

Hoare-Kalkül

$$\frac{}{\{A\}\texttt{skip}\{A\}} \qquad \frac{\{A \wedge b\}c_0\{B\} \quad \{A \wedge \neg b\}c_1\{B\}}{\{A\}\texttt{if } b \texttt{ then } c_0 \texttt{ else } c_1 \texttt{ fi}\{B\}}$$

$$\frac{}{\{B[a/X]\}X := a\{B\}} \qquad \frac{\{A \wedge b\}c\{A\}}{\{A\}\texttt{while } b \texttt{ do } c \texttt{ done}\{A \wedge \neg b\}}$$

$$\frac{\{A\}c_0\{B\} \quad \{B\}c_1\{C\}}{\{A\}c_0;\ c_1\{C\}} \qquad \frac{\models A \Rightarrow A' \quad \{A'\}c\{B'\} \quad \models B' \Rightarrow B}{\{A\}c\{B\}}$$

Eigenschaften des Hoare-Kalküls

- Der Hoare-Kalkül ist *konsistent:* wenn $\vdash \{A\}c\{B\}$, dann $\models \{A\}c\{B\}$
- Der Hoare-Kalkül ist *nicht vollständig:*
  wenn $\models \{A\}c\{B\}$, dann nicht notwendigerweise $\vdash \{A\}c\{B\}$.
  (Gödels erster Unvollständigkeitssatz)

Weakest Precondition

Defn: Für $S \in \textbf{Com}, Q \in \textbf{Assn}, I : \textbf{Intvar} \to \mathbb{N}$, die *schwächste Vorbedingung* $wp^I(S, Q)$ ist die Zusicherung, die alle Zustände beschreibt, nach deren Ausführung S in einem Zustand terminiert, in dem Q gilt:

$$wp^I(S, Q) \stackrel{def}{=} \{\sigma \in \Sigma_\perp \mid \mathcal{D}[\![S]\!]\sigma \neq \perp, \mathcal{D}[\![S]\!]\sigma \models^I Q\}$$

$$\begin{array}{rcl}
wp(\mathtt{skip}, Q) & = & Q \\
wp(c_1;\ c_2, Q) & = & wp(c_1, wp(c_2, Q)) \\
wp(X := a, Q) & = & Q[a/X] \\
wp(\mathtt{if}\ \ b\ \mathtt{then}\ \ c_1\ \mathtt{else}\ \ c_2\ \mathtt{fi}, Q) & = & b \Rightarrow wp(c_1, Q) \wedge (\neg b \Rightarrow wp(c_2, Q)) \\
wp(\mathtt{while}\ \ b\ \mathtt{do}\ \ c\ \mathtt{done}, Q) & = & \exists\, k.\, 0 \leq k.\, P_k \\
 & & \text{mit } \{P_k\} c \{P_{k+1}\}
\end{array}$$

Zusammenfassung zur Semantik

- Operational: Formalisierung der Ausführung durch syntaxgesteuerte Ableitungsregeln
  Nützlich für Sprachdefinition, Compilerverifikation

- Denotationell: Formalisierung der Bedeutung eines Programmes durch cpo's, lFixpunkte etc.
  Nützlich für Korrektheitsbeweise, Entwicklung korrekter Progr.

- Axiomatisch: Formalisierung der Ausführung und Bedeutung durch Vor- und Nachbedingungen
  Nützlich für Programmverifikation